AD-A060 568 TEXAS A AND M UNIV COLLEGE STATION INST OF STATISTICS F/G 9/2
A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJEC--ETC(U)
AUG 78 T C BAKER, R L SIELKEN N00014-78-C-0426
UNCLASSIFIED THEMIS-TR-57 NL

1 OF 4
AD
A060568

TEXAS A&M UNIVERSITY

COLLEGE STATION, TEXAS 77843

THEMIS

TEXAS A&M UNIVERSITY
PROJECT THEMIS - TR-

Technical Report 57

A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJECT SCHEDULING PROCEDURE: STATISTICAL PERT

by

Thomas C. Baker, Jr. ~~and~~ Robert L. Sielken Jr.

Aug 78

303p.

Texas A&M University
Office of Naval Research
Contract N00014-78-C-0426
Project NR047-179

ATTACHMENT I

347 380

A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJECT SCHEDULING PROCEDURE: STATISTICAL PERT

by

Thomas C. Baker, Jr. and Robert L. Sielken Jr.

THEMIS OPTIMIZATION RESEARCH PROGRAM
Technical Report No. 57
August 1978

INSTITUTE OF STATISTICS
Texas A&M University

Research conducted through the
Texas A&M Research Foundation
and sponsored by the
Office of Naval Research
Contract N00014-78-C-0426
Project NR047-179

ATTACHMENT II

A User's Guide to the Computer Implementation of the New Project Scheduling Procedure: Statistical PERT

by

Thomas C. Baker, Jr. and Robert L. Sielken Jr.

ABSTRACT

This report documents a new project scheduling procedure developed at the Institute of Statistics, Texas A&M University. The project scheduling algorithm is a five step iterative procedure capable of determining a minimum cost project schedule when the activities making up the project have durations which are random variables. The cost of an activity is assumed to be a convex piecewise linear function of the activity's mean duration. The problem is to determine the activity mean durations which both minimize the total project cost and insure that the mean (or some specified percentile) of the corresponding project completion time distribution is less than or equal to a specified project deadline. The entire distribution of the project's completion time under the minimum cost schedule is a valuable by-product. Information on the trade-off between the project's minimum cost and its specified deadline is also provided.

This report is intended as a user's guide to the new project scheduling procedure and its computer implementation. The report includes a description of the project scheduling problem, a general overview of the scheduling procedure including references to technical reports documenting the development of the procedure, and an example of the procedure's performance. The documentation of the computer implementation includes specific input

instructions; sample input and output; flowcharts; individual program descriptions; technical details concerning temporary data sets, job control language, and program interruption and restart procedures; and program listings.

TABLE OF CONTENTS

1. The Project Scheduling Problem

A project is composed of several "tasks" or "activities." These activities can be represented by arcs in a directed network. For example, a small project might consist of activities A, B, C, D, and E with the following precedence relationships:

(i) A must be completed before either C or D can be begun;

(ii) B must be completed before D can be begun; and

(iii) C and D must both be completed before E can be begun.

The corresponding network representation is shown in Figure 1. The arc labeled F does not correspond to any "real" activity but is a "dummy" activity merely representing the precedence relation that A must be completed before D can be begun. The points numbered 1, 2, ..., 5 are called nodes. In the network representation of a project the activities originating at a node can be begun only after all activities terminating at that node have been completed.

The time that it actually takes to complete an activity once that activity has been begun is called the activity's duration and is a random variable. The cost of an activity is assumed to be a convex piecewise linear function of the activity's mean duration time. An example of an activity's cost curve is given in Figure 2. In this example TIME(1) is the minimum mean duration time that can be scheduled. TIME(4) is the cheapest mean duration and hence the maximum mean duration that would be scheduled. Of course a linear cost curve is the simplest convex piecewise linear cost function. The more general piecewise behavior, however, frequently arises if there are alternative methods of performing an activity. These methods do not differ in the end result but do differ in the amount of time they take and their cost. For example, to have a mean

duration in the interval [TIME(1), TIME(2)] might require the use of a very expensive piece of special equipment while having a mean duration in the interval [TIME(2), TIME(3)] requires only specially trained personnel and having a mean duration in the interval [TIME(3), TIME(4)] just requires varying amounts of standard resources. The form of the activity duration distribution may vary from one time interval to another. For example, the activity duration distribution might be a beta distribution when the mean duration is in [TIME(1), TIME(3)] and approximately a normal distribution when the mean duration is in [TIME(3), TIME(4)].

A project schedule is a specification of each activity's mean duration. The total project cost is simply the sum of the corresponding activity costs. The time to complete the entire project is a random variable whose distribution depends upon the activity duration distributions. The objective is to determine a minimum cost project schedule such that the mean of the corresponding project completion time distribution (or some specified percentile of the project completion time distribution) is less than or equal to a specified project deadline.

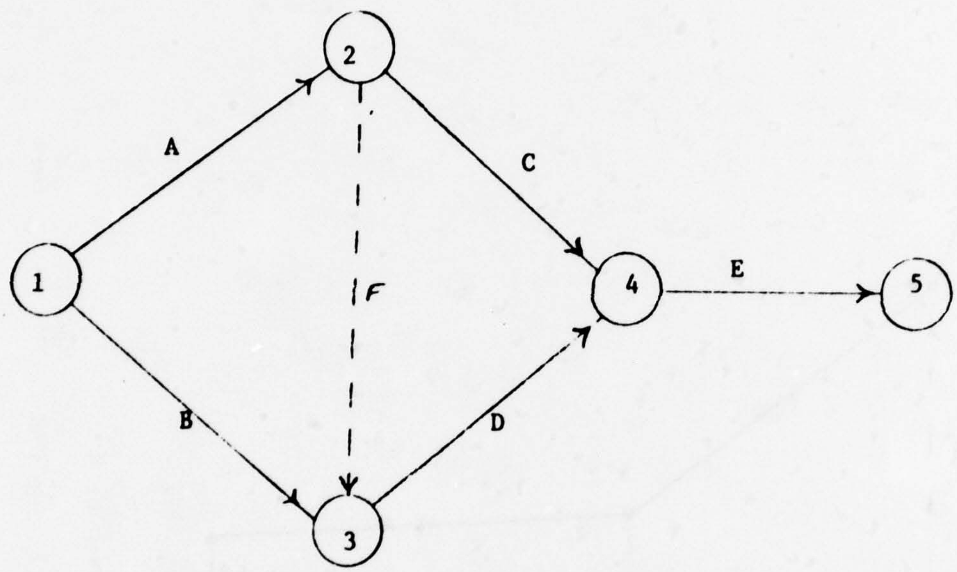


Figure 1. A Small Project Represented as a Directed Network

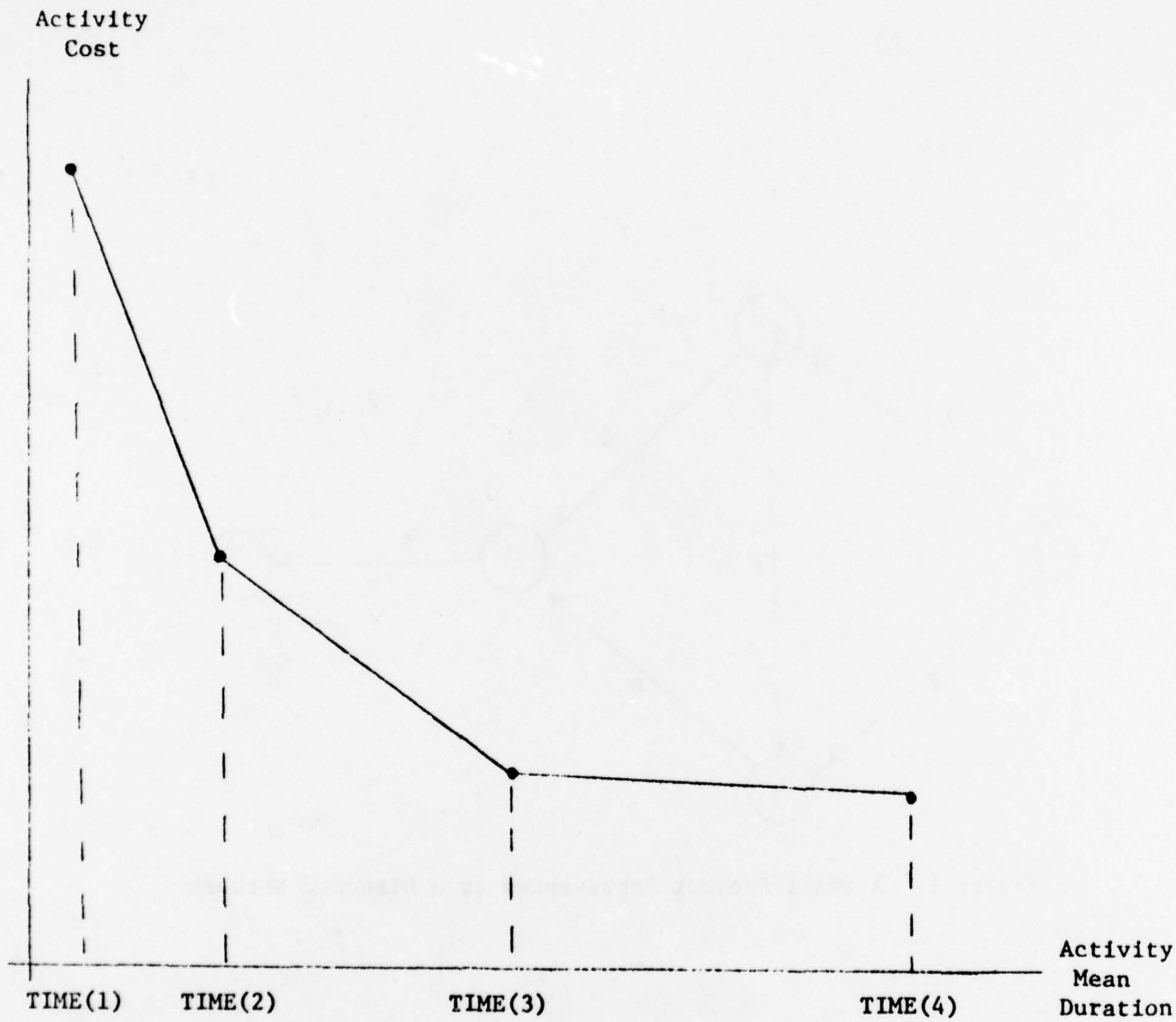


Figure 2. An Activity's Cost as a Function of the Activity's Mean Duration

2. Outline of Statistical PERT

In 1974 the development of a new project scheduling procedure was begun with the support of the Office of Naval Research. The new project scheduling procedure that has resulted is an iterative algorithm involving the following five general steps:

Step 1. Deterministic Scheduling: Find a minimum cost project schedule which completes the project by TARGET TIME when each activity's duration is exactly its mean duration and hence deterministic instead of random. (The initial value of TARGET TIME is usally the specified project deadline.)

Step 2. Simplification: Let each activity's duration be a random variable with distribution corresponding to that activity's mean duration chosen during Deterministic Scheduling. Replace various configurations of activities by single activities. The duration distribution for a replacement activity is the distribution of the time to complete all of the activities in the configuration it is replacing. The result of this step is a simplified project network with fewer activities.

Step 3. Decomposition: Partition the simplified project network into several subnetworks in such a way that the resultant subnetworks can be linked together in either series or parallel to form the simplified project network.

Step 4. Subnetwork Analysis: Analyze separately each of the subnetworks determined during Decomposition. Within a subnetwork each activity's duration distribution is approximated by a two-point discrete distribution with

matching mean, variance, and third moment. Determine the subnetwork duration distribution corresponding to these discrete activity duration distributions.

Step 5. Synthesis: Combine the approximate subnetwork duration distributions to obtain an approximate completion time distribution for the entire project. If the mean (or some specified percentile), $\hat{T}$, of this project completion time distribution is sufficiently close to the specified project deadline, the "optimal" project schedule has been found. Otherwise, reset TARGET TIME to New TARGET TIME = Old TARGET TIME* (Project Deadline/$\hat{T}$) and return to Step 1.

A general discussion and relatively nonmathematical overview of this project scheduling procedure is contained in Appendix D, "Flowchart of the Computer System," and Appendix E, "Program Descriptions"(see also Sielken and Hartley (1977)). The detailed documentation of the development thus far of each step is as follows:

Step 1. Dunn and Sielken (1977);

Step 2. Hartley and Wortham (1966) and Ringer (1969);

Step 3. Sielken and Fisher (1976);

Step 4. Sielken, Ringer, Hartley, and Arseven (1974), Sielken, Hartley, and Spoeri (1976), and Baker and Sielken (1978);

Step 5. Sielken, Ringer, Hartley and Arseven (1974) and Sielken, Hartley, and Spoeri (1976).

The forthcoming Technical Report No. 60, "Statistical PERT: The Precision of the Estimated Project Completion Time Distribution," will also be of interest to the user.

3. An Example of Statistical PERT's Performance

A small project network is depicted in Figure 3. The relationship between each activity's mean duration and its cost is given in Table 1. The project scheduling algorithm also requires that the activity's duration distribution be specified at the midpoint of each time interval on the convex piecewise linear cost function, i.e., when the activity's mean duration is [TIME(1) + TIME(2)]/2, [TIME(2) + TIME(3)]/2, etc. This information is also given in Table 1. The algorithm assumes that if an activity's mean duration is not at the midpoint but at c times the midpoint and still in the same time interval, then the activity's duration distribution has the same form (Normal, Beta, Constant, etc.) but with a new variance equal to c^2 times the variance at the midpoint. Thus, for example, if activity A's mean duration is 28, its cost is 34, and activity A's duration distribution is Beta [10,40] with mean 28 and variance $(28/25)^2 36$.

With a project deadline of 125 the algorithm's iterative determination of the minimum cost project schedule is as follows:

Step 1. Deterministic Scheduling: The shortest feasible project completion time when each activity duration is its mean duration is found to be 90 by determining the longest path through the project network when each activity's duration is equal to its minimum mean duration, TIME(1). Similarly, the longest such feasible project completion time is found to be 135 by determining the longest path through the project network when each activity's duration is equal to its maximum mean duration. The minimum cost schedule which completes the project by TARGET TIME when each activity's duration is its mean duration is determined for each value of TARGET TIME

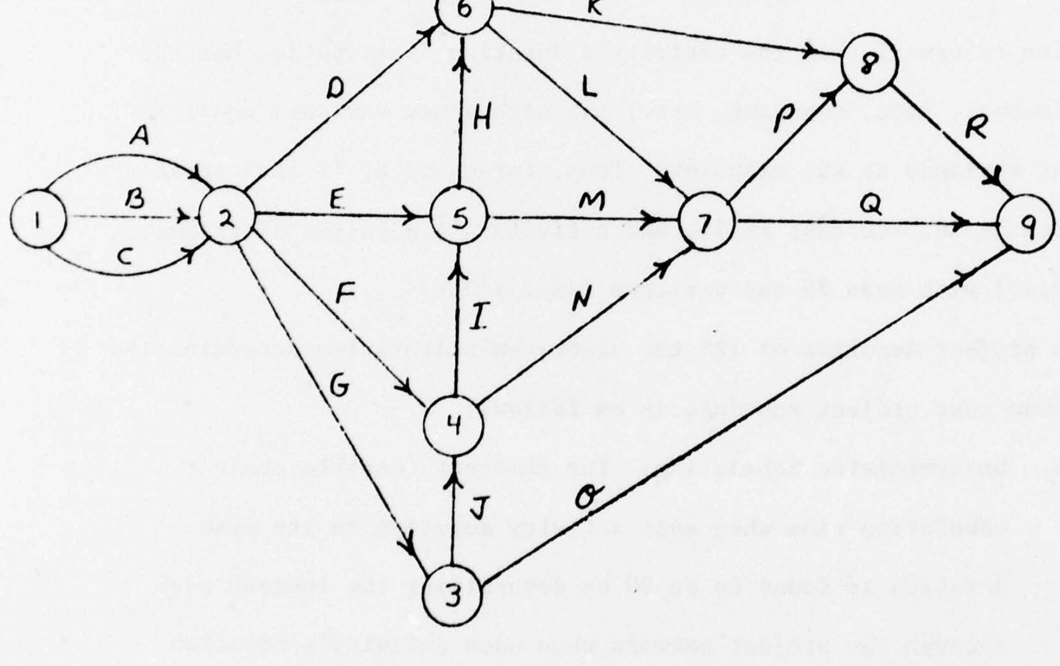


Figure 3. Example Project Network

Table 1. Activity Costs and Duration Distributions

Activity	Mean Durations	Cost	Duration Distribution at Midpoint
A	TIME(1) = 10	100	Beta on [5,20] with mean 12.5 and variance 16
	TIME(2) = 15	70	Normal with mean 17.5 and variance 25
	TIME(3) = 20	50	Beta on [10,40] with mean 25 and variance 36
	TIME(4) = 30	30	
B	TIME(1) = 8	60	Normal with mean 11 and variance 9
	TIME(2) = 14	45	
C	TIME(1) = 10	50	Constant Duration = 10
D	TIME(1) = 50	200	Normal with mean 60 and variance 100
	TIME(2) = 70	160	
E	TIME(1) = 32	45	Normal with mean 36 and variance 18
	TIME(2) = 40	40	
F	TIME(1) = 20	64	Beta on [10,50] with mean 26 and variance 70
	TIME(2) = 32	48	
G	TIME(1) = 13	30	Normal with mean 16 and variance 7
	TIME(2) = 19	25	
H	TIME(1) = 20	60	Normal with mean 25 and variance 12
	TIME(2) = 30	50	Normal with mean 32.5 and variance 9
	TIME(3) = 35	48	
I	TIME(1) = 5	60	Constant Duration = 5
J	TIME(1) = 18	62	Beta on [10,40] with mean 22 and variance 6
	TIME(2) = 26	49	
K	TIME(1) = 10	40	Constant Duration = 10
L	TIME(1) = 6	75	Normal with mean 8 and variance 5
	TIME(2) = 10	50	

Table 1. (Continued)

Activity	Mean Durations	Cost	Duration Distribution at Midpoint
M	TIME(1) = 36	225	Normal with mean 40 and variance 80
	TIME(2) = 44	175	
N	TIME(1) = 30	400	Normal with mean 35 and variance 25
	TIME(2) = 40	300	
O	TIME(1) = 60	250	Normal with mean 70 and variance 144
	TIME(2) = 80	210	
P	TIME(1) = 4	52	Constant Duration = 4
Q	TIME(1) = 2	100	Beta on [1,6] with mean 3 and variance 1
	TIME(2) = 4	90	
R	TIME(1) = 4	110	Normal with mean 5 and variance 2
	TIME(2) = 6	80	

between 90 and 135. The corresponding optimal activity mean durations are given in Table 2. The project cost curve is depicted in Figure 4.

The initial activity mean durations are those corresponding to TARGET TIME = 125; namely,

A = 30,	F = 32,	K = 10,	P = 4,
B = 14,	G = 13,	L = 10,	Q = 4,
C = 10,	H = 34,	M = 44,	R = 6.
D = 70,	I = 5,	N = 40,	
E = 40,	J = 23,	O = 80,	

Step 2. Simplification: The only configuration of activities which can be readily replaced by an equivalent single activity is A, B, and C in parallel. If the replacement activity is denoted by ABC, then the simplified project network consists of the single activity ABC and the original activities D, E, ..., R. The initial duration distribution for ABC as a function of t is

$$F_{ABC}(t) = F_A(t)F_B(t)F_C(t)$$

where $F_A(t)$ denotes a beta distribution with mean 30 and variance $(30/25)^2[(40-30)^2/36]$, $F_B(t)$ denotes a normal distribution with mean 14 and variance $(14/11)^2 9$, and $F_C(t)$ denotes the distribution for a constant duration of 10.

Step 3. Decomposition: The simplified project network is partitioned into two subnetworks in series. The first subnetwork, SUB_1, consists of the single activity ABC. The second subnetwork, SUB_2, consists of the activities D, E, ..., R.

Table 2. Optimal Activity Mean Durations for All Feasible TARGET TIME's

TARGET TIME	135-D	134-D	128-D	124-D	114-D	113-D	108-D	106-D	105-D	104-D	100-D	96-D	92-D
Range of D	$0<D\le 1$	$0<D\le 6$	$0<D\le 4$	$0<D\le 10$	$0<D\le 1$	$0<D\le 5$	$0<D\le 2$	$0<D\le 1$	$0<D\le 1$	$0<D\le 4$	$0<D\le 4$	$0<D\le 4$	$0<D\le 2$
Project Cost When D=0	1552.0	1552.4	1557.4	1563.9	1583.9	1586.1	1606.1	1614.6	1620.2	1626.2	1660.2	1702.8	1747.8
Activity													
A	30	30	30	30-D	20	20-D	15	15	15-D	14-D	10	10	10
B	14	14	14	14	14	14	14	14	14	14-D	10	10	10
C	10	10	10	10	10	10	10	10	10	10	10	10	10
D	70	70	70	70	70	70	70	70	70	70	70-D	66-D	62
E	40	40	40	40	40-D	39	39-D	37-D	36	36	36	36	36
F	32	32	32	32	32	32	32	32-D	31	31	31	31	31
G	19	19-D	13	13	13	13	13	13	13	13	13	13	13
H	35-D	34	34	34	34	34	34	34	34	34	34-D	30-D	26
I	5	5	5	5	5	5	5	5	5	5	5	5	5
J	26	26	26-D	22	22-D	21	21-D	19-D	18	18	18	18	18
K	10	10	10	10	10	10	10	10	10	10	10	10	10
L	10	10	10	10	10	10	10	10	10	10	10	10	10
M	44	44	44	44	44	44	44	44	44	44	44-D	40-D	36
N	40	40	40	40	40	40	40	40	40	40	40	40	40
O	80	80	80	80	80	80	80-D	78-D	77	77	77-D	73-D	69-D
P	4	4	4	4	4	4	4	4	4	4	4	4	4
Q	4	4	4	4	4	4	4	4	4	4	4	4	4
R	6	6	6	6	6	6	6	6	6	6	6	6	6-D

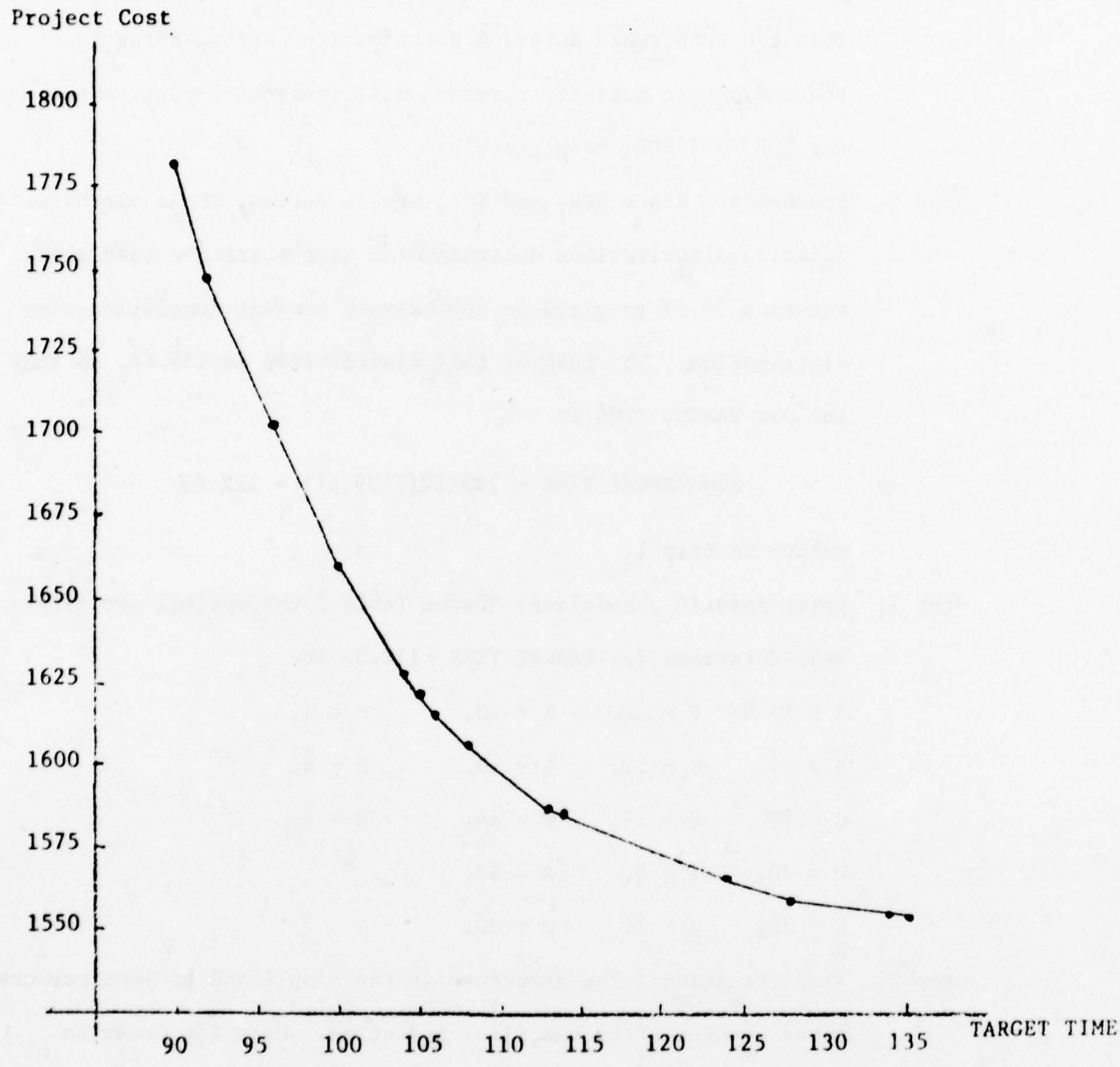


Figure 4. Total Project Cost as a Function of TARGET TIME

Step 4. Subnetwork Analysis: The activity duration distributions for activities ABC,D, ..., R are approximated by two-point discrete distributions with matching mean, variance, and third moment. Then the subnetwork duration distribution corresponding to these discrete activity duration distributions is determined for SUB_1 and SUB_2 respectively.

Step 5. Synthesis: Since SUB_1 and SUB_2 are in series, their approximate duration distributions determined in Step 4 are combined using equation (8.1) to yield an approximate project completion time distribution. The mean of this distribution is 139.42, so that the new TARGET TIME is

$$\text{New TARGET TIME} = 125(125/139.42) = 112.07$$

Return to Step 1.

Step 1. Deterministic Scheduling: Using Table 2 the optimal activity mean durations for TARGET TIME = 112.09 are

A = 19.07	F = 32,	K = 10,	P = 4,
B = 14,	G = 13,	L = 10,	Q = 4,
C = 10,	H = 34,	M = 44,	R = 6,
D = 70,	I = 5,	N = 40,	
E = 39,	J = 21,	O = 80,	

Step 2. Simplification: The structure of the simplified project network never changes after the first iteration. Only the duration distribution of ABC needs to be redetermined.

Step 3. Decomposition: The partitioning of the simplified project network never changes after the first iteration.

Step 4. Subnetwork Analysis: The new activity duration distributions are approximated by new two-point discrete distributions. Then new approximate duration distributions for SUB_1 and SUB_2 are determined.

Step 5. Synthesis: The new approximate project mean completion time is 128.42, so that the new TARGET TIME is

$$\text{New TARGET TIME} = 112.07(125/128.42) = 109.09.$$

Return to Step 1.

If 128.42 was sufficiently close to the specified project deadline of 125, then these optimal activity mean durations would constitute the minimum cost project schedule. Otherwise, the project scheduling algorithm could be continued.

4. Concluding Remarks

The new project scheduling procedure allows the project scheduler to specify

(i) the precedences among the project's activities,

(ii) the relationship between an activity's cost and its mean duration,

(iii) the manner in which an activity's actual duration varies about its mean duration, and

(iv) a deadline for either the project's mean completion time or a prescribed percentile of the project completion time distribution.

In return the project scheduler receives

(i) a minimum cost project schedule which delineates each activity's mean duration time,

(ii) an estimate of the <u>distribution</u> of the project completion time,

(iii) information on the trade-off between the project's minimum cost and its specified deadline, and

(iv) a tool for monitoring the project's progress and, if need be, rescheduling.

An exciting feature of this new project scheduling procedure is that it simultaneously incorporates the desire to minimize the project cost and the realization that an activity's duration is not necessarily a fixed quantity exactly equal to its prescribed duration but rather a random quantity varying about a prescribed duration. No longer must the project scheduler either (i) choose a reasonable cost schedule which heuristically hedges against the randomness in the activities he guesses will be critical, or (ii) choose a reasonable schedule which should probably finish before the deadline and then guess where he can save money without disturbing the suspected completion time too much. By considering both cost and randomness

together in one systematic algorithm, the new project scheduling procedure eliminates this guesswork.

The authors wish to acknowledge their gratitude to the Office of Naval Research for the support of this research under contracts N00014-68-A-0140 and N00014-76-C-0038. Several present and past members of the Institute of Statistics at Texas A&M University have also contributed to the development of this new project scheduling procedure and its computer implementation: E. Arseven (Lederle Laboratories), P. P. Biemer, C. S. Dunn, N. E. Fisher (Compucon Inc.), L. J. Ringer, and R. K. Spoeri (Bureau of the Census). The authors also want to particularly acknowledge the considerable contributions of H. O. Hartley to Statistical PERT.

References

Baker, Thomas C. Jr. and R. L. Sielken Jr., "Statistical PERT: Improvements in the Determination of the Project Completion Time Distribution", THEMIS Technical Report 58, August 1978.

Baker, Thomas C. Jr. and R. L. Sielken Jr., "Statistical PERT: The Precision of the Estimated Project Completion Time Distribution", THEMIS Project Technical Report No. 60, August 1978.

Biemer, Paul P., and R. L. Sielken Jr., "Incorporating Project Cost Considerations into Stochastic PERT", THEMIS Project Technical Report No. 52, November, 1975.

Dunn, C. S. and R. L. Sielken Jr., "Statistical PERT: An Improved Project Scheduling Algorithm", THEMIS Project Technical Report No. 55, February, 1977.

Fulkerson, D. R., "A Network Flow Computation for Project Cost Curves", Management Science, Vol. 7, No. 2, (January, 1961), pp. 167-178.

Hartley, H. O. and A. W. Wortham, "A Statistical Theory for PERT Critical Path Analysis," Management Science, Vol. 12, No. 10, (June, 1966), pp. 469-481.

Ringer, L. J., "Numerical Operators for Statistical PERT Critical Path Analysis", Management Science, Vol. 16, No. 2, (October, 1969), pp. B-136 - B-143.

Sielken, R. L. Jr. and Norman E. Fisher, "Statistical PERT: Decomposing a Project Network", THEMIS Project Technical Report No. 50, January, 1976.

Sielken, R. L. Jr. and H. O. Hartley, "A New Statistical Approach to Project Scheduling", THEMIS Project Technical Report No. 56, December 1977.

Sielken, R. L. Jr., H. O. Hartley and R. K. Spoeri, "Statistical PERT: An Improved Subnetwork Analysis Procedure", THEMIS Project Technical Report No. 51, January 1976.

Sielken, R. L. Jr., L. J. Ringer, H. O. Hartley and E. Arseven, "Statistical Critical Path Analysis in Acyclic Stochastic Networks: Statistical PERT", THEMIS Project Technical Report No. 48, November, 1974.

Spoeri, R. K., L. J. Ringer and R. L. Sielken Jr., "A Statistical Procedure for Optimization of PERT Network Scheduling Systems", THEMIS Project Technical Report No. 53, April, 1976.

Appendix A

Specific Input Instructions for the Statistical PERT Computer System

Only the main program (MAIN) requires any input data cards. The order and setup of those cards is indicated below.

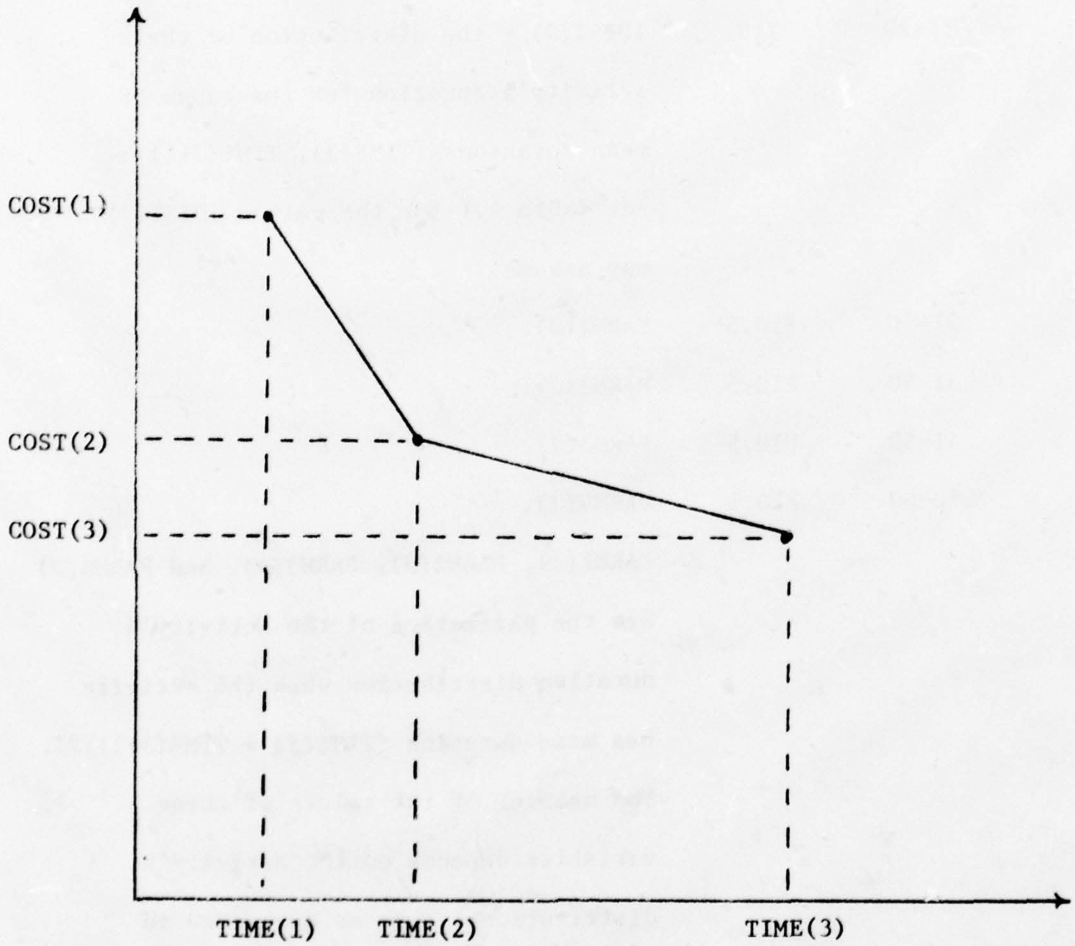
Card Order	Column	Format	Description
1	1-10	I10	NACT = the number of activities in the project network: NACT $\leq$ 1000.
	11-20	I10	NODES = the number of nodes in the project network; NODES $\leq$ 1000.
	21-30	I10	NSRCE = the number of the project's source node; 1 $\leq$ NSRCE $\leq$ 9999.
	31-40	I10	NSINK = the number of the project's sink node; 1 $\leq$ NSINK $\leq$ 9999.
	41-50	I10	LNODEN = the largest node number in the project network; LNODEN $\leq$ 9999.

* *

Repeat cards 2 - 5 for each activity in the project network.

Card Order	Column	Format	Description
2	1-10	I10	I = the number of the activity being described; 1 $\leq$ I $\leq$ 9999. (Note: The network's activities need not be numbered consecutively, and they need not be described in order.)
	11-20	I10	NODEO = the activity's origin node number; 1 $\leq$ NODEO $\leq$ 9999.
	21-30	I10	NODET = the activity's terminal node number; 1 $\leq$ NODET $\leq$ 9999.

	31-40	I10	NCT = the number of completion times and costs needed to specify this activity's piecewise linear cost curve as a function of its mean duration; $2 \leq NCT \leq 6$. See Figure A.1 for clarification. (Note: For activities with constant or fixed duration, NCT $\equiv$ 2.)
3	1-10	I10	TIME(1).
	11-20	I10	TIME(2).
	⋮		⋮
	(NCT-1)*10+1-NCT*10	I10	TIME(NCT)
			TIME(J) = the J-th activity completion time; $0 \leq TIME(J) < 32768 = 2^{15}$. These times must be in increasing order. See Figure A.1 for clarification. (Note: For activities with constant duration TIME(1) $\equiv$ TIME(2).)
4	1-10	I10	COST(1).
	11-20	I10	COST(2).
	⋮		⋮
	(NCT-1)*10+1-NCT*10	I10	COST(NCT).
			COST(J) = the cost associated with the J-th activity completion time; $0 \leq COST(J) < 32768 = 2^{15}$. See Figure A.1 for clarification. (Note: For activities with constant duration, COST(1) $\equiv$ COST(2).)



(NCT = 3)

Figure A.1

An example of an activity's cost curve.

Card 5 is a "packet" composed of NCT - 1 cards each containing the following information.

5	1-10	I10	J = 1, 2, ..., NCT - 1.
	11-20	I10	IDIST(J) = the distribution of the activity's duration for the range of mean durations (TIME(J), TIME(J+1)); see Table A.1 for the values IDIST(J) may assume.
	21-30	F10.5	PARM1(J).
	31-40	F10.5	PARM2(J).
	41-50	F10.5	PARM3(J).
	50-60	F10.5	PARM4(J).
			PARM1(J), PARM2(J), PARM3(J), and PARM4(J) are the parameters of the activity's duration distribution when the activity has mean duration (TIME(J) + TIME(J+1))/2. The meaning of the values of these variables depends on the activity's distributional type as described in Table A.1.

* *

6	1-10	I10	TEST1.
	11-20	I10	TEST2.
			TEST1 and TEST2 control the amount of output from the deterministic project scheduling program (DPS).

Table A.1

Activity Duration Distributions and Their Parameters

Distribution	Beta		Normal		Rectangular	Dummy
IDIST	-1		0		1	2
PARM1	Min Time	Min Time	Min Time	....	Min Time	Fixed Time
PARM2	Mode	Alpha, α	Mean, μ	Mean, μ	....	....
PARM3	Max Time	Max Time	Max Time	St Dev., σ	Max Time	....
PARM4	-1	Beta, β	-1	1	....	....
	a	b	c	d	e	f

Interpreted Distribution:

a) $$f(x) = \frac{\Gamma(a+b+2)(x-\text{Min})^a(\text{Max}-x)^b}{\Gamma(a+1)\Gamma(b+1)(\text{Max}-\text{Min})^{a+b+1}} \qquad \text{Min} \leq x \leq \text{Max}$$

where a, b chosen so that $\mu = \dfrac{\text{Min}+4*\text{Mode}+\text{Max}}{6}$ and $\sigma^2 = \dfrac{(\text{Max}-\text{Min})^2}{36}$

b) $$f(x) = \frac{\Gamma(\alpha+\beta+2)(x-\text{Min})^\alpha(\text{Max}-x)^\beta}{\Gamma(\alpha+1)\Gamma(\beta+1)(\text{Max}-\text{Min})^{\alpha+\beta+1}}, \qquad \text{Min} \leq x \leq \text{Max}, \quad \alpha > -1 \quad \beta > -1$$

which implies $\mu = \dfrac{\text{Min}(\beta+1)+\text{Max}(\alpha+1)}{(\alpha+\beta+2)}$

c) $$f(x) = \frac{1}{\sqrt{2\pi}\, s} e^{-\frac{1}{2}\left(\frac{x-\mu}{s}\right)^2},$$

where $s = (\text{Max} - \text{Min})/6$ and $\mu \geq 3s$ is reasonable

d) $$f(x) = \frac{1}{\sqrt{2\pi}\, \sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2},$$

where $\mu \geq 3\sigma$ is reasonable

e) $$f(x) = \frac{1}{(\text{Max}-\text{Min})}, \qquad \text{Min} \leq x \leq \text{Max}$$

f) $$f(x) = 1, \qquad x = \text{FIXED TIME}$$

TEST1 = input data option.

TEST1 = 0 implies all the information input to the program is to be printed.

TEST1 = 1 implies the input data is not to be printed.

TEST2 = intermediate output option.

TEST2 = 0 implies the output arising from intermediate calculations is to be printed in addition to the optimal activity mean durations for every feasible completion time.

TEST2 = 1 implies the optimal activity mean durations for every feasible project completion time are to be printed but the intermediate calculations are not to be printed.

21-30	I10	JMAT = an output option in the simplification program (SIMP).

JMAT = 1 implies that only the activity number, origin node, terminal node, and duration distribution for

for each activity in the simplified network is to be printed (minimum output).

JMAT = 0 implies the minimum output plus a listing of which activities were simplified and what activity configurations are to be printed (complete output).

The quantities on card 7 are all parameters used by the subnetwork analysis program (SUBNET). See Appendix D for a more detailed description of Subnetwork Analysis.

7	1-10	I10	IEDF = the number of subdivisions in the estimated duration distribution for each subnetwork; $2 \leq$ IEDF ≤ 20.
	11-20	I10	NMAX = the maximum number of activities in a subnetwork for which the subnetwork's entire discrete duration distribution is to be automatically enumerated; since 2^{NMAX} calculations are required, a reasonable value for NMAX is ≤ 12.
	21-30	I10	IPOOL = the bound computation option. IPOOL = 0 implies the maximum-cluster procedure is to be used to determine the

upper and lower bounds (F_2^+, F_2^-) on each subnetwork's discrete duration distribution.

IPOOL = 1 implies the union-cluster procedure is to be used to determine the upper and lower bounds (F_1^+, F_2^-) on each subnetwork's discrete duration distribution.

31-40 I10 SAMSIZ = the maximum number of activity duration configurations to be explicitly considered in the determination of the upper and lower bounds on each subnetwork's discrete duration distribution; SAMSIZ $\geq 2^{NMAX}$.

Card 8 contains additional parameters used by the subnetwork analysis program (SUBNET).

8 1-10 I10 THELAM = (θ,λ) pair option.

THELAM = 0 implies the standard (θ,λ) pairs are to be used when determining clusters, i.e., (θ,λ) = (1,1), (2,2), (3,2).

THELAM = 1 implies the three specified (θ,λ) pairs are to be used when determining clusters instead of the standard pairs.

	11-20	F10.5	THETA(1).
	21-30	F10.5	THETA(2).
	31-40	F10.5	THETA(3).
	41-50	F10.5	LAMBDA(1).
	51-60	F10.5	LAMBDA(2).
	61-70	F10.5	LAMBDA(3).

THETA(I) = the I-th θ-value to be used if THELAM = 1.

LAMBDA(I) = the I-th λ-value to be used if THELAM = 1.

The following relations must always be true:

$$0 \leq \text{THETA}(1) \leq \text{THETA}(2) \leq \text{THETA}(3)$$

$$0 \leq \text{LAMBDA}(1) \leq \text{LAMBDA}(2) \leq \text{LAMBDA}(3).$$

If THELAM = 0, these values are ignored.

The quantities on card 9 are parameters used by the synthesis program (SYNTH).

9	1-10	I10	IOPT = a computation option.

IOPT = 0 implies the PCT-th percentile of the network's approximate

		completion time distribution is to be compared to the specified project deadline, PD.
		IOPT = 1 implies the mean of the network's approximate completion time distribution is to be compared to the specified project deadline, PD.
11-20	I10	NT = the number of subdivisions in the network's synthesized completion time distribution; NT $\leq$ 49.
21-30	F10.5	PCT = the percentile value to be used if IOPT = 0; for example, PCT = 90.0.
31-40	F10.5	PD = the specified project deadline.
41-50	F10.5	TT = the initial target time to be used by the deterministic project scheduler (DPS) in attempting to generate a project schedule whose corresponding completion time distribution has mean (IOPT = 1) or PCT-th percentile (IOPT = 0) equal equal to PD. Usually TT $\leq$ PD works best. A standard value would be TT = PD.

APPENDIX B

Computer Input for the Example in Section 3

18	9	1	9	9	
1	1	2	4		
10	15	20	30		
100	70	50	30		
1	-1	5.	.257812	20.	.257812
2	0		17.5	5.	1.
3	-1	10.	25.	40.	-1.
2	1	2	2		
8	14				
60	45				
1	0		11.	3.	1.
3	1	2	2		
10	10				
50	50				
1	2	10.			
4	2	6	2		
50	70				
200	160				
1	0		60.	10.	1.
5	2	5	2		
32	40				
45	40				
1	0		36.	4.242641	1.
6	2	4	2		
20	32				
64	48				
1	-1	10.	.794285	50.	1.691428
7	2	3	2		
13	19				
30	25				
1	0		16.	2.645751	1.
8	5	6	3		
20	30	35			

60	50	48			
1	0		25.	3.464102	1.
2	0	23.5	32.5	41.5	-1.
9	4	5	2		
5	5				
60	60				
1	2	5.			
10	3	4	2		
18	26				
62	49				
1	-1	10.	13.	40.	20.
11	6	8	2		
10	10				
40	40				
1	2	10.			
12	6	7	2		
6	10				
75	50				
1	0		8.	2.236068	1.
13	5	7	2		
36	44				
225	175				
1	0		40.	8.944272	1.
14	4	7	2		
30	40				
400	300				
1	0	20.	35.	50.	-1.
15	3	9	2		
60	80				
250	210				
1	0		70.	12.	1.
16	7	8	2		
4	4				
52	52				
1	2	4.			
17	7	9	2		
2	4				
100	90				
1	-1	1.	1.	6.	2.
18	8	9	2		
4	6				
110	80				
1	0		5.	1.414214	1.
1	1	1			
20	9	1	500		
0					
1	49		125.	125.	

Appendix C

Computer Output for the Example in Section 3

```
                         S T A T I S T I C A L   P E R T

                         A PROJECT SCHEDULING PROCEDURE

                                 WRITTEN AT THE

                            INSTITUTE OF STATISTICS
                             TEXAS A&M UNIVERSITY
                         COLLEGE STATION, TEXAS 77843

                                  AUGUST 1978

          INQUIRIES AND COMMENTS SHOULD BE ADDRESSED TO: ROBERT L. SIELKEN JR.


THIS IS THE OUTPUT FROM THE MAIN PROGRAM: MAIN


THE PROJECT NETWORK AS IT WAS READ IN:


THE NUMBER OF NODES IS     9.
THE NUMBER OF ACTIVITIES IS   18.
THE SOURCE NODE IS NUMBERED      1 AND THE SINK NODE IS NUMBERED     9.
THE LARGEST NODE NUMBER IS      9.
```

ACTIVITY	ORIG	TERM	J	TIME	COST	DISTRIBUTION	PARAMETERS
1	1	2	1	10	100	BETA	MIN= 5.00000, MAX= 20.00000, ALPHA= 0.25781, BETA= 0.25781
1	1	2	2	15	70	NORMAL	MEAN= 17.50000, ST.DEV= 5.00000
1	1	2	3	20	50	BETA	MIN= 10.00000, MODE= 25.00000, MAX= 40.00000
1	1	2	4	30	30		
2	1	2	1	8	60	NORMAL	MEAN= 11.00000, ST.DEV= 3.00000
2	1	2	2	14	45		
3	1	2	1	10	50	FIXED(CONSTANT)	TIME= 10.00000
3	1	2	2	10	50		
4	2	6	1	50	200	NORMAL	MEAN= 60.00000, ST.DEV= 10.00000
4	2	6	2	70	160		
5	2	5	1	32	45	NORMAL	MEAN= 36.00000, ST.DEV= 4.24264
5	2	5	2	49	40		
6	2	4	1	20	64	BETA	MIN= 10.00000, MAX= 50.00000, ALPHA= 0.79428, BETA= 1.69143
6	2	4	2	32	48		
7	2	3	1	13	30	NORMAL	MEAN= 16.00000, ST.DEV= 2.64575
7	2	3	2	19	25		
8	5	6	1	20	60	NORMAL	MEAN= 25.00000, ST.DEV= 3.46410
8	5	6	2	30	50	NORMAL	MIN= 23.50000, MEAN= 32.50000, MAX= 41.50000
8	5	6	3	35	48		
9	4	5	1	5	60	FIXED(CONSTANT)	TIME= 5.00000
9	4	5	2	5	60		
10	3	4	1	18	62	BETA	MIN= 10.00000, MAX= 40.00000, ALPHA= 13.00000, BETA= 20.00000
10	3	4	2	26	49		
11	6	8	1	10	40	FIXED(CONSTANT)	TIME= 10.00000
11	6	8	2	10	40		
12	6	7	1	6	75	NORMAL	MEAN= 8.00000, ST.DEV= 2.23607
12	6	7	2	10	50		
13	5	7	1	36	225	NORMAL	MEAN= 40.00000, ST.DEV= 8.94427
13	5	7	2	44	175		
14	4	7	1	30	400	NORMAL	MIN= 20.00000, MEAN= 35.00000, MAX= 50.00000
14	4	7	2	40	300		
15	3	9	1	60	250	NORMAL	MEAN= 70.00000, ST.DEV= 12.00000
15	3	9	2	80	210		
16	7	8	1	4	52	FIXED(CONSTANT)	TIME= 4.00000
16	7	8	2	4	52		
17	7	9	1	2	100	BETA	MIN= 1.00000, MAX= 6.00000, ALPHA= 1.00000, BETA= 2.00000
17	7	9	2	4	90		
18	8	9	1	4	110	NORMAL	MEAN= 5.00000, ST.DEV= 1.41421
18	8	9	2	6	80		

THE INFORMATION INPUT TO THE DETERMINISTIC PROJECT SCHEDULING PROGRAM WILL NOT BE PRINTED.
THE INTERMEDIATE OUTPUT GENERATED BY THE DETERMINISTIC PROJECT SCHEDULING PROGRAM WILL NOT BE PRINTED.

THE SIMPLIFICATION PROGRAM WILL PRINT ONLY THE MINIMUM OUTPUT.

DURING THE SUBNETWORK ANALYSIS PROGRAM,
THE NUMBER OF SUBDIVISIONS IN THE ESTIMATED DURATION DISTRIBUTION FOR EACH SUBNETWORK WILL BE 20;
THE LARGEST NUMBER OF ACTIVITIES IN A SUBNETWORK FOR WHICH THE SUBNETWORK'S EXACT DISCRETE DURATION DISTRIBUTION WILL BE ENUMERATED IS 9;
THE MAXIMUM NUMBER OF ACTIVITY DURATION CONFIGURATIONS EXPLICITLY CONSIDERED IN THE DETERMINATION OF THE UPPER AND LOWER BOUNDS ON EACH SUBNETWORK'S DISCRETE DURATION DISTRIBUTION WILL BE 500;
THE UNION-CLUSTER PROCEDURE WILL BE USED IN THE DETERMINATION OF THE BOUNDS;
THE FOLLOWING (THETA,LAMBDA) PAIRS WILL BE USED IN THE FORMATION OF THE CLUSTERS:
(1.00000, 1.00000),(2.00000, 2.00000),(3.00000, 2.00000).

```
THE NUMBER OF SUBDIVISIONS IN THE NETWORK'S SYNTHESIZED COMPLETION TIME DISTRIBUTION WILL BE 49.

THE MEAN OF THE NETWORK'S APPROXIMATE COMPLETION TIME DISTRIBUTION WILL BE COMPARED TO THE
         SPECIFIED PROJECT DEADLINE OF   125.00000.

THE TARGET TIME FOR THE FIRST ITERATION IS  125.00000.


THIS IS THE OUTPUT FROM SUBPROGRAM LOOP. THE PROGRAM SEARCHES THE GIVEN PROJECT NETWORK FOR LOOPS (CYCLES).
A VALID PROJECT NETWORK SHOULD CONTAIN NO LOOPS.


NODE     1 IS NOT PART OF A LOOP
NODE     2 IS NOT PART OF A LOOP
NODE     3 IS NOT PART OF A LOOP
NODE     4 IS NOT PART OF A LOOP
NODE     5 IS NOT PART OF A LOOP
NODE     6 IS NOT PART OF A LOOP
NODE     7 IS NOT PART OF A LOOP
NODE     8 IS NOT PART OF A LOOP
NODE     9 IS NOT PART OF A LOOP
THERE ARE NO LOOPS IN THIS NETWORK
```

```
*****************************************************************************************************************
*****************************************************************************************************************

THIS IS THE OUTPUT FROM THE DETERMINISTIC PROJECT SCHEDULER: DPS

*****************************************************************************************************************
*****************************************************************************************************************

THE ENTIRE PROJECT COST CURVE WILL BE DETERMINED.

LAMBDA = PROJECT COMPLETION TIME

THE STARTING VALUE OF LAMBDA IS          135.

THE CORRESPONDING TOTAL PROJECT COST IS        0.15520E 04.

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    1.
LAMBDA RANGES FROM        135 TO        134.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        135-D:

PROJECT COMPLETION TIME =        135-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1                     30                 0.30000E 02
        2                     14                 0.45000E 02
        3                     10                 0.50000E 02
        4                     70                 0.16000E 03
        5                     40                 0.40000E 02
        6                     32                 0.48000E 02
        7                     19                 0.25000E 02
        8                     35-D               0.48000E 02 + (  0.40000E 00*D)
        9                      5                 0.60000E 02
       10                     26                 0.49000E 02
       11                     10                 0.40000E 02
       12                     10                 0.50000E 02
       13                     44                 0.17500E 03
       14                     40                 0.30000E 03
       15                     80                 0.21000E 03
       16                      4                 0.52000E 02
       17                      4                 0.90000E 02
       18                      6                 0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS        0.15520E 04 + (  0.40015E 00*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    6.
LAMBDA RANGES FROM        134 TO        128.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        134-D:

PROJECT COMPLETION TIME =        134-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1                     30                 0.30000E 02
        2                     14                 0.45000E 02
        3                     10                 0.50000E 02
        4                     70                 0.16000E 03
        5                     40                 0.40000E 02
```

```
         6                  32              0.48000E 02
         7                  19-D            0.25000E 02 + (  0.83333E 00*D)
         8                  34              0.48400E 02
         9                   5              0.60000E 02
        10                  26              0.49000E 02
        11                  10              0.40000E 02
        12                  10              0.50000E 02
        13                  44              0.17500E 03
        14                  40              0.30000E 03
        15                  80              0.21000E 03
        16                   4              0.52000E 02
        17                   4              0.90000E 02
        18                   6              0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS     0.15524E 04 + (  0.83333E 00*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO   4.
LAMBDA RANGES FROM       128 TO        124.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        128-D:

PROJECT COMPLETION TIME =        128-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)         ACTIVITY COST
         1                  30              0.30000E 02
         2                  14              0.45000E 02
         3                  10              0.50000E 02
         4                  70              0.16000E 03
         5                  40              0.40000E 02
         6                  32              0.48000E 02
         7                  13              0.30000E 02
         8                  34              0.48400E 02
         9                   5              0.60000E 02
        10                  26-D            0.49000E 02 + (  0.16250E 01*D)
        11                  10              0.40000E 02
        12                  10              0.50000E 02
        13                  44              0.17500E 03
        14                  40              0.30000E 03
        15                  80              0.21000E 03
        16                   4              0.52000E 02
        17                   4              0.90000E 02
        18                   6              0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS     0.15574E 04 + (  0.16250E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO  10.
LAMBDA RANGES FROM       124 TO        114.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        124-D:

PROJECT COMPLETION TIME =        124-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)         ACTIVITY COST
         1                  30-D            0.30000E 02 + (  0.20000E 01*D)
         2                  14              0.45000E 02
         3                  10              0.50000E 02
         4                  70              0.16000E 03
         5                  40              0.40000E 02
         6                  32              0.48000E 02
```

```
       7                 13              0.30000E 02
       8                 34              0.48400E 02
       9                  5              0.60000E 02
      10                 22              0.55500E 02
      11                 10              0.40000E 02
      12                 10              0.50000E 02
      13                 44              0.17500E 03
      14                 40              0.30000E 03
      15                 80              0.21000E 03
      16                  4              0.52000E 02
      17                  4              0.90000E 02
      18                  6              0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS     0.15639E 04 + (  0.20000E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    1.
LAMBDA RANGES FROM         114 TO      113.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        114-D:

PROJECT COMPLETION TIME =         114-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
       1                 20              0.50000E 02
       2                 14              0.45000E 02
       3                 10              0.50000E 02
       4                 70              0.16000E 03
       5                 40-D            0.40000E 02 + (  0.62500E 00*D)
       6                 32              0.48000E 02
       7                 13              0.30000E 02
       8                 34              0.48400E 02
       9                  5              0.60000E 02
      10                 22-D            0.55500E 02 + (  0.16250E 01*D)
      11                 10              0.40000E 02
      12                 10              0.50000E 02
      13                 44              0.17500E 03
      14                 40              0.30000E 03
      15                 80              0.21000E 03
      16                  4              0.52000E 02
      17                  4              0.90000E 02
      18                  6              0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS     0.15839E 04 + (  0.22500E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    5.
LAMBDA RANGES FROM         113 TO      108.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =        113-D:

PROJECT COMPLETION TIME =         113-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
       1                 20-D            0.50000E 02 + (  0.40000E 01*D)
       2                 14              0.45000E 02
       3                 10              0.50000E 02
       4                 70              0.16000E 03
       5                 39              0.40625E 02
       6                 32              0.48000E 02
       7                 13              0.30000E 02
```

```
        8                    34                 0.48400E 02
        9                     5                 0.60000E 02
       10                    21                 0.57125E 02
       11                    10                 0.40000E 02
       12                    10                 0.50000E 02
       13                    44                 0.17500E 03
       14                    40                 0.30000E 03
       15                    80                 0.21000E 03
       16                     4                 0.52000E 02
       17                     4                 0.90000E 02
       18                     6                 0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS      0.15861E 04 + (  0.40000E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    2.
LAMBDA RANGES FROM         108 TO        106.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =          108-D:

PROJECT COMPLETION TIME =        108-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1                    15                 0.70000E 02
        2                    14                 0.45000E 02
        3                    10                 0.50000E 02
        4                    70                 0.16000E 03
        5                    39-D               0.40625E 02 + (  0.62500E 00*D)
        6                    32                 0.48000E 02
        7                    13                 0.30000E 02
        8                    34                 0.48400E 02
        9                     5                 0.60000E 02
       10                    21-D               0.57125E 02 + (  0.16250E 01*D)
       11                    10                 0.40000E 02
       12                    10                 0.50000E 02
       13                    44                 0.17500E 03
       14                    40                 0.30000E 03
       15                    80-D               0.21000E 03 + (  0.20000E 01*D)
       16                     4                 0.52000E 02
       17                     4                 0.90000E 02
       18                     6                 0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS      0.16061E 04 + (  0.42500E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    1.
LAMBDA RANGES FROM         106 TO        105.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =          106-D:

PROJECT COMPLETION TIME =        106-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1                    15                 0.70000E 02
        2                    14                 0.45000E 02
        3                    10                 0.50000E 02
        4                    70                 0.16000E 03
        5                    37-D               0.41875E 02 + (  0.62500E 00*D)
        6                    32-D               0.48000E 02 + (  0.13333E 01*D)
        7                    13                 0.30000E 02
        8                    34                 0.48400E 02
```

```
        9                  5             0.60000E 02
       10               19-D             0.60375E 02 + (  0.16250E 01*D)
       11                 10             0.40000E 02
       12                 10             0.50000E 02
       13                 44             0.17500E 03
       14                 40             0.30000E 03
       15               78-D             0.21400E 03 + (  0.20000E 01*D)
       16                  4             0.52000E 02
       17                  4             0.90000E 02
       18                  6             0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS       0.16146E 04 + (  0.55833E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    1.
LAMBDA RANGES FROM        105 TO      104.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =          105-D:

PROJECT COMPLETION TIME =          105-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1               15-D             0.70000E 02 + (  0.60000E 01*D)
        2                 14             0.45000E 02
        3                 10             0.50000E 02
        4                 70             0.16000E 03
        5                 36             0.42500E 02
        6                 31             0.49333E 02
        7                 13             0.30000E 02
        8                 34             0.48400E 02
        9                  5             0.60000E 02
       10                 18             0.62000E 02
       11                 10             0.40000E 02
       12                 10             0.50000E 02
       13                 44             0.17500E 03
       14                 40             0.30000E 03
       15                 77             0.21600E 03
       16                  4             0.52000E 02
       17                  4             0.90000E 02
       18                  6             0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS       0.16202E 04 + (  0.60000E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO    4.
LAMBDA RANGES FROM        104 TO      100.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =          104-D:

PROJECT COMPLETION TIME =          104-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
        1               14-D             0.76000E 02 + (  0.60000E 01*D)
        2               14-D             0.45000E 02 + (  0.25000E 01*D)
        3                 10             0.50000E 02
        4                 70             0.16000E 03
        5                 36             0.42500E 02
        6                 31             0.49333E 02
        7                 13             0.30000E 02
        8                 34             0.48400E 02
        9                  5             0.60000E 02
```

```
      10                    18                  0.62000E 02
      11                    10                  0.40000E 02
      12                    10                  0.50000E 02
      13                    44                  0.17500E 03
      14                    40                  0.30000E 03
      15                    77                  0.21600E 03
      16                     4                  0.52000E 02
      17                     4                  0.90000E 02
      18                     6                  0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS      0.16262E 04 + (  0.85000E 01*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO   4.
LAMBDA RANGES FROM         100 TO        96.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =          100-D:

PROJECT COMPLETION TIME =          100-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
       1                    10                  0.10000E 03
       2                    10                  0.55000E 02
       3                    10                  0.50000E 02
       4                    70-D                0.16000E 03 + (  0.20000E 01*D)
       5                    36                  0.42500E 02
       6                    31                  0.49333E 02
       7                    13                  0.30000E 02
       8                    34-D                0.48400E 02 + (  0.40000E 00*D)
       9                     5                  0.60000E 02
      10                    18                  0.62000E 02
      11                    10                  0.40000E 02
      12                    10                  0.50000E 02
      13                    44-D                0.17500E 03 + (  0.62500E 01*D)
      14                    40                  0.30000E 03
      15                    77-D                0.21600E 03 + (  0.20000E 01*D)
      16                     4                  0.52000E 02
      17                     4                  0.90000E 02
      18                     6                  0.80000E 02

THE CURRENT VALUE OF THE PROJECT COST IS      0.16602E 04 + (  0.10650E 02*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO   4.
LAMBDA RANGES FROM          96 TO        92.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE =           96-D:

PROJECT COMPLETION TIME =           96-D.

  ACTIVITY #: I    NEW VALUE: XACT(I)          ACTIVITY COST
       1                    10                  0.10000E 03
       2                    10                  0.55000E 02
       3                    10                  0.50000E 02
       4                    66-D                0.16800E 03 + (  0.20000E 01*D)
       5                    36                  0.42500E 02
       6                    31                  0.49333E 02
       7                    13                  0.30000E 02
       8                    30-D                0.50000E 02 + (  0.10000E 01*D)
       9                     5                  0.60000E 02
      10                    18                  0.62000E 02
```

11	10	0.40000E 02	
12	10	0.50000E 02	
13	40-D	0.20000E 03 + (	0.62500E 01*D)
14	40	0.30000E 03	
15	73-D	0.22400E 03 + (	0.20000E 01*D)
16	4	0.52000E 02	
17	4	0.90000E 02	
18	6	0.80000E 02	

THE CURRENT VALUE OF THE PROJECT COST IS 0.17028E 04 + (0.11250E 02*D).

DELTA (REPRESENTED BY "D") RANGES FROM 0 TO 2.
LAMBDA RANGES FROM 92 TO 90.
THE MINIMUM COST PROJECT SCHEDULE FOR PROJECT DEADLINE = 92-D:

PROJECT COMPLETION TIME = 92-D.

ACTIVITY #: I	NEW VALUE: XACT(I)	ACTIVITY COST	
1	10	0.10000E 03	
2	10	0.55000E 02	
3	10	0.50000E 02	
4	62	0.17600E 03	
5	36	0.42500E 02	
6	31	0.49333E 02	
7	13	0.30000E 02	
8	26	0.54000E 02	
9	5	0.60000E 02	
10	18	0.62000E 02	
11	10	0.40000E 02	
12	10	0.50000E 02	
13	36	0.22500E 03	
14	40	0.30000E 03	
15	69-D	0.23200E 03 + (	0.20000E 01*D)
16	4	0.52000E 02	
17	4	0.90000E 02	
18	6-D	0.80000E 02 + (	0.15000E 02*D)

THE CURRENT VALUE OF THE PROJECT COST IS 0.17478E 04 + (0.17000E 02*D).

* * * * *

THE SINK WAS REACHED WITH INFINITE CAPACITY IMPLYING AN INFEASIBLE SOLUTION TO THE PRIMAL PROBLEM
IF LAMBDA DROPS BELOW ITS CURRENT VALUE. 90.

```
************************************************************************************************************
************************************************************************************************************

THIS IS THE OUTPUT FROM THE DETERMINISTIC SCHEDULE RESOLUTION PROGRAM: DSR

************************************************************************************************************
************************************************************************************************************
```

THE MINIMUM COST PROJECT SCHEDULE FOR A TARGET TIME OF 125.00000 IS AS FOLLOWS:

ACTIVITY:	MEAN	DISTRIBUTION	PARAMETERS							
1	30.00000	BETA	MIN=	12.00000,	MAX=	48.00000,	ALPHA=	3.00000,	BETA=	3.00000
2	14.00000	NORMAL	MEAN=	14.00000,	ST.DEV=	3.81818				
3	10.00000	FIXED(CONSTANT)	TIME=	10.00000						
4	70.00000	NORMAL	MEAN=	70.00000,	ST.DEV=	11.66666				
5	40.00000	NORMAL	MEAN=	40.00000,	ST.DEV=	4.71404				
6	32.00000	BETA	MIN=	12.30770,	MAX=	61.53848,	ALPHA=	0.79428,	BETA=	1.69143
7	13.00000	NORMAL	MEAN=	13.00000,	ST.DEV=	2.14967				
8	34.00000	NORMAL	MEAN=	34.00000,	ST.DEV=	3.13846				
9	5.00000	FIXED(CONSTANT)	TIME=	5.00000						
10	23.00000	BETA	MIN=	10.45455,	MAX=	41.81821,	ALPHA=	12.99998,	BETA=	20.00002
11	10.00000	FIXED(CONSTANT)	TIME=	10.00000						
12	10.00000	NORMAL	MEAN=	10.00000,	ST.DEV=	2.79508				
13	44.00000	NORMAL	MEAN=	44.00000,	ST.DEV=	9.83869				
14	40.00000	NORMAL	MEAN=	40.00000,	ST.DEV=	5.71428				
15	80.00000	NORMAL	MEAN=	80.00000,	ST.DEV=	13.71428				
16	4.00000	FIXED(CONSTANT)	TIME=	4.00000						
17	4.00000	BETA	MIN=	1.33333,	MAX=	8.00000,	ALPHA=	1.00000,	BETA=	2.00000
18	6.00000	NORMAL	MEAN=	6.00000,	ST.DEV=	1.69705				

```
************************************************************************************************************************************
************************************************************************************************************************************
THIS IS THE OUTPUT FROM THE SIMPLIFICATION PROGRAM: SIMP
************************************************************************************************************************************
************************************************************************************************************************************
```

THE FOLLOWING ACTIVITIES CANNOT BE FURTHER COMBINED:

ACTIVITY	ORIGIN	TERMINAL	ACTIVITY NUMBER ASSIGNED BY DECOMP
2	1	2	1
4	2	6	2
5	2	5	3
6	2	4	4
7	2	3	5
8	5	6	6
9	4	5	7
10	3	4	8
11	6	8	9
12	6	7	10
13	5	7	11
14	4	7	12
15	3	9	13
16	7	8	14
17	7	9	15
18	8	9	16

```
*****************************************************************************************************************************
*****************************************************************************************************************************

THIS IS THE OUTPUT FROM THE MODIFIED SIMPLIFICATION PROGRAM: MODSIMP

*****************************************************************************************************************************
*****************************************************************************************************************************


   ACTIVITY      ORIGIN      TERMINAL      MEAN       STANDARD DEVIATION
      2             1            2         30.8              6.40

                         F(T)        T
                         0.0  --    15.6000
                         0.05 --    20.0877
                         0.15 --    23.3456
                         0.25 --    25.5891
                         0.35 --    27.4363
                         0.45 --    29.1490
                         0.55 --    33.6000
                         0.65 --    33.6000
                         0.75 --    37.2000
                         0.85 --    37.2000
                         0.95 --    40.8000
                         1.00 --    44.4000


   ACTIVITY      ORIGIN      TERMINAL      MEAN       STANDARD DEVIATION
      4             2            6         70.0             10.94

                         F(T)        T
                         0.0  --    35.0000
                         0.05 --    50.8092
                         0.15 --    57.9081
                         0.25 --    62.1310
                         0.35 --    65.5046
                         0.45 --    68.5340
                         0.55 --    71.4660
                         0.65 --    74.4954
                         0.75 --    77.8690
                         0.85 --    82.0919
                         0.95 --    89.1908
                         1.00 --   105.0000
```

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
5	2	5	40.0	4.42

F(T)	T
0.0 --	25.8579
0.05 --	32.2457
0.15 --	35.1141
0.25 --	36.8204
0.35 --	38.1836
0.45 --	39.4076
0.55 --	40.5924
0.65 --	41.8164
0.75 --	43.1796
0.85 --	44.8859
0.95 --	47.7543
1.00 --	54.1421

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
6	2	4	31.9	10.03

F(T)	T
0.0 --	12.3077
0.05 --	16.5614
0.15 --	20.5854
0.25 --	23.8162
0.35 --	26.8016
0.45 --	29.7300
0.55 --	32.7243
0.65 --	35.9089
0.75 --	39.4651
0.85 --	43.7664
0.95 --	50.1376
1.00 --	61.5385

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
7	2	3	13.0	2.02

F(T)	T
0.0 --	6.5510
0.05 --	9.4639
0.15 --	10.7720
0.25 --	11.5501
0.35 --	12.1717
0.45 --	12.7299
0.55 --	13.2701
0.65 --	13.8283
0.75 --	14.4499
0.85 --	15.2280
0.95 --	16.5361
1.00 --	19.4490

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
8	5	6	34.0	2.94

F(T)	T
0.0 --	24.5846
0.05 --	28.8375
0.15 --	30.7471
0.25 --	31.8831
0.35 --	32.7907
0.45 --	33.6056
0.55 --	34.3944
0.65 --	35.2093
0.75 --	36.1169
0.85 --	37.2529
0.95 --	39.1625
1.00 --	43.4154

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
9	4	5	5.0	0.00

F(T)	T
0.0 --	5.0000
0.05 --	5.0000
0.15 --	5.0000
0.25 --	5.0000
0.35 --	5.0000
0.45 --	5.0000
0.55 --	5.0000
0.65 --	5.0000
0.75 --	5.0000
0.85 --	5.0000
0.95 --	5.0000
1.00 --	5.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
10	3	4	23.0	2.42

F(T)	T
0.0 --	10.4546
0.05 --	18.8855
0.15 --	20.3169
0.25 --	21.2104
0.35 --	21.9427
0.45 --	22.6122
0.55 --	23.2690
0.65 --	23.9550
0.75 --	24.7250
0.85 --	25.6542
0.95 --	27.3216
1.00 --	41.8182

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
11	6	8	10.0	0.00

F(T)	T
0.0 --	10.0000
0.05 --	10.0000
0.15 --	10.0000
0.25 --	10.0000
0.35 --	10.0000
0.45 --	10.0000
0.55 --	10.0000
0.65 --	10.0000
0.75 --	10.0000
0.85 --	10.0000
0.95 --	10.0000
1.00 --	10.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
12	6	7	10.0	2.62

F(T)	T
0.0 --	1.6147
0.05 --	5.4023
0.15 --	7.1030
0.25 --	8.1147
0.35 --	8.9230
0.45 --	9.6488
0.55 --	10.3512
0.65 --	11.0770
0.75 --	11.8653
0.85 --	12.8970
0.95 --	14.5977
1.00 --	18.3853

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
13	5	7	44.0	9.23

F(T)	T
0.0 --	14.4839
0.05 --	27.8160
0.15 --	33.8027
0.25 --	37.3639
0.35 --	40.2090
0.45 --	42.7637
0.55 --	45.2363
0.65 --	47.7910
0.75 --	50.6361
0.85 --	54.1973
0.95 --	60.1840
1.00 --	73.5161

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
14	4	7	40.0	5.36

F(T)		T
0.0	--	22.8572
0.05	--	30.6004
0.15	--	34.0774
0.25	--	36.1458
0.35	--	37.7982
0.45	--	39.2819
0.55	--	40.7181
0.65	--	42.2018
0.75	--	43.8542
0.85	--	45.9226
0.95	--	49.3996
1.00	--	57.1428

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
15	3	9	80.0	12.86

F(T)		T
0.0	--	38.8572
0.05	--	57.4410
0.15	--	65.7858
0.25	--	70.7499
0.35	--	74.7156
0.45	--	78.2767
0.55	--	81.7233
0.65	--	85.2844
0.75	--	89.2501
0.85	--	94.2142
0.95	--	102.5590
1.00	--	121.1428

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
16	7	8	4.0	0.00

F(T)		T
0.0	--	4.0000
0.05	--	4.0000
0.15	--	4.0000
0.25	--	4.0000
0.35	--	4.0000
0.45	--	4.0000
0.55	--	4.0000
0.65	--	4.0000
0.75	--	4.0000
0.85	--	4.0000
0.95	--	4.0000
1.00	--	4.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
17	7	9	4.0	1.30

F(T)		T
0.0	—	1.3333
0.05	--	1.9840
0.15	--	2.5292
0.25	--	2.9535
0.35	--	3.3398
0.45	--	3.7155
0.55	--	4.0979
0.65	--	4.5038
0.75	--	4.9579
0.85	--	5.5103
0.95	—	6.3427
1.00	--	8.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
18	8	9	6.0	1.59

F(T)		T
0.0	—	0.9088
0.05	—	3.2085
0.15	--	4.2411
0.25	—	4.8554
0.35	--	5.3461
0.45	—	5.7667
0.55	—	6.2133
0.65	—	6.6539
0.75	—	7.1446
0.85	--	7.7589
0.95	--	8.7915
1.00	—	11.0912

```
*********************************************************************************************************
*********************************************************************************************************

THIS IS THE OUTPUT FROM THE DECOMPOSITION PROGRAM: DECOMP

*********************************************************************************************************
*********************************************************************************************************

    INPUT STAGE

    THE SIMPLIFIED NETWORK HAS              16 ARCS

    THE SOURCE IS NODE NUMBER            1

    THE SINK IS NODE NUMBER              9

    THE LARGEST NODE IS NODE NUMBER      9

    THE SIMPLIFIED NETWORK AS READ IN IS:

    ARC NUMBER     ORIGIN NODE     TERMINAL NODE
         1               1               2
         2               2               6
         3               2               5
         4               2               4
         5               2               3
         6               5               6
         7               4               5
         8               3               4
         9               6               8
        10               6               7
        11               5               7
        12               4               7
        13               3               9
        14               7               8
        15               7               9
        16               8               9
```

```
STAGE  1 BREAKUP

SUBNETWORK   1 IS COMPOSED OF SUBNETWORKS:

  2.  3.

IN SERIES
```

```
STAGE  2 BREAKUP

          SUBNETWORK   2 IS A NON-DECOMPOSABLE NETWORK
          IT IS COMPOSED OF:

              SOURCE NODE =   1

              SINK NODE   =   2

          THE SUBNETWORK AS IT IS READ IN:

              ARC  S(ARC)  T(ARC)
               1     1       2

          THE FOLLOWING REPRESENTATION OF THE SUBNETWORK ABOVE WILL BE USED BY THE SUBNETWORK ANALYSIS PROGRAM:

              ACTIVITY    TAIL       HEAD
                 1          1          2
```

```
SUBNETWCRK   3 IS A NON-DECOMPOSABLE NETWORK
IT IS CCMPCSED CF:

    SOURCE NCDE =   2

    SINK NCDE   =   9

THE SUBNETWCRK AS IT IS READ IN:
```

ARC	S(ARC)	T(ARC)
2	2	6
3	2	5
4	2	4
5	2	3
9	6	8
10	6	7
6	5	6
11	5	7
7	4	5
12	4	7
8	3	4
13	3	9
16	8	9
14	7	8
15	7	9

```
THE FCLLCWING REPRESENTATION OF THE SUBNETWORK ABOVE WILL BE USED BY THE SUBNETWORK ANALYSIS PROGRAM:
```

ACTIVITY	TAIL	HEAD
1	1	5
2	1	4
3	1	3
4	1	2
5	5	7
6	5	6
7	4	5
8	4	6
9	3	4
10	3	6
11	2	3
12	2	8
13	7	8
14	6	7
15	6	8

THE NUMBER OF NON-DECOMPOSABLE SUBNETWORKS IS 2

```
************************************************************************************************************************************
************************************************************************************************************************************

THIS IS THE OUTPUT FROM THE SUBNETWORK ANALYSIS PROGRAM: SUBNET

************************************************************************************************************************************
************************************************************************************************************************************


EACH SUBNETWORK'S APPROXIMATE DURATION DISTRIBUTION IS DETERMINED.
```

SUBNETWORK 2

INITIAL INPUT

ACTIVITY	ORIGIN	TERMINAL	LOWER POINT	UPPER POINT	MEAN	STANDARD DEVIATION	PROB. LOWER PT.
1	1	2	24.0785	36.8868	30.8008	6.3962	0.47516

THE DURATION DISTRIBUTION FOR SUBNETWORK 2

TIME	DISTRIBUTION
0.15600D 02	0.0
0.20088D 02	0.50000D-01
0.23346D 02	0.15000D 00
0.25589D 02	0.25000D 00
0.27436D 02	0.35000D 00
0.29149D 02	0.45000D 00
0.33600D 02	0.55000D 00
0.33600D 02	0.65000D 00
0.37200D 02	0.75000D 00
0.37200D 02	0.85000D 00
0.40800D 02	0.95000D 00
0.44400D 02	0.10000D 01

SUBNETWORK 3

INITIAL INPUT

ACTIVITY	ORIGIN	TERMINAL	LOWER POINT	UPPER POINT	MEAN	STANDARD DEVIATION	PROB. LOWER PT.
1	1	5	59.0567	80.9433	70.0000	10.9433	0.50000
2	1	4	35.5782	44.4218	40.0000	4.4218	0.50000
3	1	3	23.0185	43.2217	31.9497	10.0336	0.55793
4	1	2	10.9836	15.0164	13.0000	2.0164	0.50000
5	5	7	10.0000	10.0000	10.0000	0.0	1.00000
6	5	6	7.3782	12.6218	10.0000	2.6218	0.50000
7	4	5	31.0561	36.9439	34.0000	2.9439	0.50000
8	4	6	34.7713	53.2287	44.0000	9.2287	0.50000
9	3	4	5.0000	5.0000	5.0000	0.0	1.00000
10	3	6	34.6400	45.3600	40.0000	5.3600	0.50000
11	2	3	20.6713	25.5124	22.9933	2.4185	0.52037
12	2	8	67.1360	92.8640	80.0000	12.8640	0.50000
13	7	8	4.4082	7.5918	6.0000	1.5918	0.50000
14	6	7	4.0000	4.0000	4.0000	0.0	1.00000
15	6	8	2.8330	5.4402	3.9934	1.2957	0.55491

THE CRDER IN WHICH TO CONSIDER THE ACTIVITIES TO DETERMINE THE LONGEST PATH TIME:

ORDER	ACTIVITY
1	1
2	2
3	3
4	4
5	11
6	12
7	9
8	10
9	7
10	8
11	5
12	6
13	14
14	15
15	13

THE CRITICAL PATH TIME WHEN EACH ACTIVITY'S COMPLETION TIME IS SET EQUAL TO ITS MEAN IS = 0.94993D 02

THE 8 NODES ON THE CRITICAL PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
8. 7. 6. 5. 4. 3. 2. 1.

THE 7 CRITICAL ACTIVITIES ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
13. 14. 6. 7. 9. 11. 4.

A LOWER BOUND ON THE SUBNETWORK DURATION TIME IS = 0.83497D 02

THE 8 NODES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
8. 7. 6. 5. 4. 3. 2. 1.

THE 7 ACTIVITIES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
13. 14. 6. 7. 9. 11. 4.

A UPPER BOUND ON THE SUBNETWORK DURATION TIME IS = 0.11304D 03

THE 6 NODES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
8. 7. 6. 4. 3. 1.

THE 5 ACTIVITIES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
13. 14. 8. 9. 3.

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.10000D 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  13. IS =   0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  14. IS =   0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    3-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   6. IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    4-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   7. IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    5-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   9. IS =   0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    6-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  11. IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    7-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   4. IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

THERE ARE    5 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE    1-TH CLUSTER ARE AS FOLLOWS:
          13.

     THE ACTIVITIES IN THE    2-TH CLUSTER ARE AS FOLLOWS:
          14.

     THE ACTIVITIES IN THE    3-TH CLUSTER ARE AS FOLLOWS:
           6.   8.   7.

     THE ACTIVITIES IN THE    5-TH CLUSTER ARE AS FOLLOWS:
           9.

     THE ACTIVITIES IN THE    6-TH CLUSTER ARE AS FOLLOWS:
          11.   2.   4.

     THERE ARE    6 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =      0.10000D 01

     THERE ARE    8 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
            1.       1
            2.       2
            3.       3
            4.       5
            5.       8
            6.      10
            7.      12
            8.      15

     THERE ARE    4 ELIMINANTS CORRESPONDING TO ACTIVITY    1
        THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    7
        THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    9
        THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY   11
        THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    4

     THERE ARE    3 ELIMINANTS CORRESPONDING TO ACTIVITY    2
        THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY    9
        THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY   11
        THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY    4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    3
        THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY   11
        THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY    4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    8
        THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY    6
        THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY    7

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   10

     THERE ARE    6 ELIMINANTS CORRESPONDING TO ACTIVITY   12
        THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   13
        THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   14
        THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    6
        THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    7
        THE    5-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    9
        THE    6-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   11

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   15
```

```
THERE ARE    1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

     THERE ARE  12 ACTIVITIES IN THE   1-TH CLUSTER.  THEY ARE AS FOLLOWS:
                13. 12. 14.  6.  8.  7.  1.  9. 11.  2.  4.  3.

          7 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FOLLOWS:
                 1.  2.  3.  4.  5.  6.  7.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA,LAMBDA) = (0.10000D 01 , 0.10000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.85646D 02	0.23221D-02	0.23221D-02
0.87795D 02	0.23221D-02	0.69664D-02
0.89944D 02	0.11611D-01	0.13751D-01
0.92092D 02	0.18213D-01	0.18213D-01
0.94241D 02	0.20536D-01	0.20536D-01
0.96390D 02	0.22858D-01	0.30374D-01
0.98538D 02	0.75167D-01	0.80398D-01
0.10069D 03	0.11936D 00	0.11936D 00
0.10284D 03	0.18818D 00	0.18818D 00
0.10498D 03	0.35409D 00	0.35409D 00
0.10713D 03	0.50588D 00	0.50588D 00
0.10779D 03	0.52300D 00	0.52300D 00
0.10845D 03	0.67962D 00	0.67962D 00
0.10910D 03	0.67962D 00	0.67962D 00
0.10976D 03	0.75374D 00	0.75374D 00
0.11042D 03	0.88884D 00	0.88884D 00
0.11107D 03	0.88884D 00	0.88884D 00
0.11173D 03	0.88884D 00	0.88884D 00
0.11239D 03	0.88884D 00	0.88884D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.20000D 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  13, IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          12.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  14, IS =   0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    3-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   6, IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    4-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   7, IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    5-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   9, IS =   0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    6-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  11, IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    7-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   4, IS =   1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

THERE ARE    5 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE    1-TH CLUSTER ARE AS FOLLOWS:
          13. 12.

     THE ACTIVITIES IN THE    2-TH CLUSTER ARE AS FOLLOWS:
          14.

     THE ACTIVITIES IN THE    3-TH CLUSTER ARE AS FOLLOWS:
           6.  8.  7.

     THE ACTIVITIES IN THE    5-TH CLUSTER ARE AS FOLLOWS:
           9.

     THE ACTIVITIES IN THE    6-TH CLUSTER ARE AS FOLLOWS:
          11.  2.  4.

     THERE ARE    5 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =      0.20000D 01

     THERE ARE    8 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
            1.      1
            2.      2
            3.      3
            4.      5
            5.      8
            6.     10
            7.     12
            8.     15

     THERE ARE    4 ELIMINANTS CORRESPONDING TO ACTIVITY    1
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    7
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    9
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY   11
         THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY    4

     THERE ARE    3 ELIMINANTS CORRESPONDING TO ACTIVITY    2
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY    9
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY   11
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY    4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    3
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY   11
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY    4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    8
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY    6
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY    7

     THERE ARE    3 ELIMINANTS CORRESPONDING TO ACTIVITY   10
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   10 IS ACTIVITY    6
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   10 IS ACTIVITY    7
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY   10 IS ACTIVITY    9

     THERE ARE    6 ELIMINANTS CORRESPONDING TO ACTIVITY   12
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   13
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   14
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    6
         THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    7
         THE    5-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY    9
         THE    6-TH ELIMINANT CORRESPONDING TO ACTIVITY   12 IS ACTIVITY   11

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   15
```

```
THERE ARE    1 NCNEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

     THERE ARE  13 ACTIVITIES IN THE    1-TH CLUSTER.  THEY ARE AS FOLLOWS:
               13. 12. 14.  6.  8.  7.  1.  9. 11.  2.  4.  3. 10.

           7 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FOLLOWS:
               1.  2.  3.  4.  5.  6.  7.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA,LAMBDA) = (0.20000D 01 , 0.20000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.85646D 02	0.23213D-02	0.23213D-02
0.87795D 02	0.69638D-02	0.69638D-02
0.89944D 02	0.92850D-02	0.92850D-02
0.92092D 02	0.11606D-01	0.11606D-01
0.94241D 02	0.24989D-01	0.24989D-01
0.96390D 02	0.34861D-01	0.34861D-01
0.98538D 02	0.81552D-01	0.81552D-01
0.10069D 03	0.13154D 00	0.13154D 00
0.10284D 03	0.18928D 00	0.18928D 00
0.10498D 03	0.34177D 00	0.34177D 00
0.10713D 03	0.49998D 00	0.49998D 00
0.10779D 03	0.52137D 00	0.52137D 00
0.10845D 03	0.64707D 00	0.64707D 00
0.10910D 03	0.64707D 00	0.64707D 00
0.10976D 03	0.73837D 00	0.73837D 00
0.11042D 03	0.88491D 00	0.88491D 00
0.11107D 03	0.88491D 00	0.88491D 00
0.11173D 03	0.88491D 00	0.88491D 00
0.11239D 03	0.88491D 00	0.88491D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.20000D 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  13. IS =  1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          12.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  14. IS =  0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    3-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   6. IS =  1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    4-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   7. IS =  1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           8.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    5-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   9. IS =  0

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    6-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  11. IS =  1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    7-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   4. IS =  1

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
           2.

THERE ARE    5 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE    1-TH CLUSTER ARE AS FOLLOWS:
          13. 12.

     THE ACTIVITIES IN THE    2-TH CLUSTER ARE AS FOLLOWS:
          14.

     THE ACTIVITIES IN THE    3-TH CLUSTER ARE AS FOLLOWS:
           6.  8.  7.

     THE ACTIVITIES IN THE    5-TH CLUSTER ARE AS FOLLOWS:
           9.

     THE ACTIVITIES IN THE    6-TH CLUSTER ARE AS FOLLOWS:
          11.  2.  4.

     THERE ARE    5 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =      0.300000 01

     THERE ARE    8 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
            1.       1
            2.       2
            3.       3
            4.       5
            5.       8
            6.      10
            7.      12
            8.      15

     THERE ARE    4 ELIMINANTS CORRESPONDING TO ACTIVITY   1
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   1 IS ACTIVITY    7
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   1 IS ACTIVITY    9
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY   1 IS ACTIVITY   11
         THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY   1 IS ACTIVITY    4

     THERE ARE    3 ELIMINANTS CORRESPONDING TO ACTIVITY   2
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   2 IS ACTIVITY    9
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   2 IS ACTIVITY   11
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY   2 IS ACTIVITY    4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY   3
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   3 IS ACTIVITY   11
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   3 IS ACTIVITY    4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY   8
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   8 IS ACTIVITY    6
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   8 IS ACTIVITY    7

     THERE ARE    3 ELIMINANTS CORRESPONDING TO ACTIVITY  10
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY  10 IS ACTIVITY    6
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY  10 IS ACTIVITY    7
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY  10 IS ACTIVITY    9

     THERE ARE    6 ELIMINANTS CORRESPONDING TO ACTIVITY  12
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY   13
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY   14
         THE    3-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY    6
         THE    4-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY    7
         THE    5-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY    9
         THE    6-TH ELIMINANT CORRESPONDING TO ACTIVITY  12 IS ACTIVITY   11

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY  15
```

```
THERE ARE    1 NCNEMPTY CLLSTERS AFTER POOLING CN THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

     THERE ARE  13 ACTIVITIES IN THE    1-TH CLUSTER.  THEY ARE AS FOLLOWS:
               13. 12. 14.  6.  8.  7.  1.  9. 11.  2.  4.  3. 10.

            7 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FCLLOWS:
               1.  2.  3.  4.  5.  6.  7.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA,LAMBDA) = (0.30000D 01 , 0.20000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.85646D 02	0.23213D-02	0.23213D-02
0.87795D 02	0.69638D-02	0.69638D-02
0.89944D 02	0.92850D-02	0.92850D-02
0.92092D 02	0.11606D-01	0.11606D-01
0.94241D 02	0.24989D-01	0.24989D-01
0.96390D 02	0.34861D-01	0.34861D-01
0.98538D 02	0.81552D-01	0.81552D-01
0.10069D 03	0.13154D 00	0.13154D 00
0.10284D 03	0.18928D 00	0.18928D 00
0.10498D 03	0.34177D 00	0.34177D 00
0.10713D 03	0.49998D 00	0.49998D 00
0.10779D 03	0.52137D 00	0.52137D 00
0.10845D 03	0.64707D 00	0.64707D 00
0.10910D 03	0.64707D 00	0.64707D 00
0.10976D 03	0.73837D 00	0.73837D 00
0.11042D 03	0.88491D 00	0.88491D 00
0.11107D 03	0.88491D 00	0.88491D 00
0.11173D 03	0.88491D 00	0.88491D 00
0.11239D 03	0.88491D 00	0.88491D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

THE UPPER AND LOWER BOUNDS ON THE DISCRETE SUBNETWORK DURATION DISTRIBUTION ARE EQUAL.

THUS AN ESTIMATE OF THE DISCRETE DURATION DISTRIBUTION FOR SUBNETWORK 3 IS

TIME	DISTRIBUTION
0.856460D 02	0.232130D-02
0.877950D 02	0.696380D-02
0.899440D 02	0.928500D-02
0.920920D 02	0.116060D-01
0.942410D 02	0.249890D-01
0.963900D 02	0.348610D-01
0.985380D 02	0.815520D-01
0.100690D 03	0.131540D 00
0.102840D 03	0.189280D 00
0.104980D 03	0.341770D 00
0.107130D 03	0.499980D 00
0.107790D 03	0.521370D 00
0.108450D 03	0.647070D 00
0.109100D 03	0.647070D 00
0.109760D 03	0.738370D 00
0.110420D 03	0.884910D 00
0.111070D 03	0.884910D 00
0.111730D 03	0.884910D 00
0.112390D 03	0.884910D 00
0.113040D 03	0.100000D 01

```
****************************************************************************************************
****************************************************************************************************

THIS IS THE OUTPUT FROM THE SYNTHESIS PROGRAM: SYNTH

****************************************************************************************************
****************************************************************************************************


THE CURRENT PROJECT SCHEDULE IS THE ONE MOST RECENTLY LISTED BY THE DETERMINISTIC SCHEDULE RESOLUTION PROGRAM.

THE PROJECT'S CORRESPONDING APPROXIMATE COMPLETION TIME DISTRIBUTION IS DETERMINED BELOW:


FOR THIS PROBLEM

THE NUMBER OF C.D.F. SUBDIVISIONS USED THROUGHOUT SYNTHESIS IS   49.
THE TARGET TIME USED BY THE DETERMINISTIC SCHEDULER WAS  125.00000
THE MEAN OF THE PROJECT'S APPROXIMATE COMPLETION TIME DISTRIBUTION WILL BE COMPARED TO THE
SPECIFIED PROJECT DEADLINE TIME OF     125.00000

        THE INSTRUCTIONS TO BE PERFORMED ARE

                                   SERIES=0
    INSTRUCTION NO.    SUBNETWORK    PARALLEL=1     SUBNETWORKS TO BE SYNTHESIZED

          1                1             0              2  3
```

SUBNETWORK	APPROXIMATE DISTRIBUTION DETERMINED USING SUBNETWORK ANALYSIS	
2	X	F(X)
	15.6000	0.0
	20.0876	0.05000
	23.3455	0.15000
	25.5891	0.25000
	27.4362	0.35000
	29.1490	0.45000
	33.6000	0.55000
	33.6000	0.65000
	37.2000	0.75000
	37.2000	0.85000
	40.8000	0.95000
	44.4000	1.00000
3	X	F(X)
	85.6461	0.00232
	87.7948	0.00696
	89.9435	0.00929
	92.0922	0.01161
	94.2409	0.02499
	96.3896	0.03486
	98.5384	0.08155
	100.6871	0.13154
	102.8358	0.18928
	104.9845	0.34177
	107.1332	0.49998
	107.7898	0.52137
	108.4463	0.64707
	109.1029	0.64707
	109.7594	0.73837
	110.4160	0.88491
	111.0725	0.88491
	111.7291	0.88491
	112.3856	0.88491
	113.0422	1.00000

THE DISTRIBUTION FOR THE SYNTHESIZED NETWORK IS

T	F(T)
[illegible]	0.0
[illegible]	0.0
[illegible]	0.0
[illegible]	0.0
105.83355	0.00012
106.98041	0.00012
108.12727	0.00035
109.27413	0.00058
110.42099	0.00070
111.56785	0.00139
112.71471	0.00151
113.86157	0.00244
115.00843	0.00334
116.15529	0.00427
117.30215	0.00522
118.44901	0.00703
119.59587	0.01029
120.74273	0.01262
121.88959	0.02228
123.03645	0.02662
124.18331	0.03508
125.33017	0.04763
126.47703	0.06476
127.62389	0.07313
128.77075	0.11432
129.91761	0.12435
131.06447	0.17084
132.21133	0.20357
133.35819	0.24976
134.50505	0.30646
135.65191	0.33489
136.79876	0.41044
137.94562	0.45992
139.09248	0.51173
140.23934	0.54260
141.38620	0.58624
142.53306	0.66266
143.67992	0.68903
144.82678	0.74598
145.97364	0.79715
147.12050	0.83843
148.26736	0.88644
149.41422	0.90878
150.56108	0.94093
151.70794	0.96349
152.85480	0.97084
154.00166	0.98235
155.14852	0.99425
156.29538	0.99425
157.44224	1.00000

THE MEAN OF THE SYNTHESIZED DISTRIBUTION IS 139.41685

THE DIFFERENCE BETWEEN 139.41685 AND THE PROJECT DEADLINE IS 11.53 PERCENT OF THE PROJECT DEADLINE.
HENCE. THE NEW TARGET TIME IS 112.07397

```
************************************************************************************************************
THIS COMPLETES ITERATICN        1
************************************************************************************************************
```

THIS IS THE OUTPUT FROM THE DETERMINISTIC SCHEDULE RESOLUTION PROGRAM: DSR

THE MINIMUM COST PROJECT SCHEDULE FOR A TARGET TIME OF 112.07396 IS AS FOLLOWS:

ACTIVITY:	MEAN	DISTRIBUTION	PARAMETERS
1	19.07396	NORMAL	MEAN= 19.07396, ST.DEV= 5.44970
2	14.00000	NORMAL	MEAN= 14.00000, ST.DEV= 3.81818
3	10.00000	FIXED(CONSTANT)	TIME= 10.00000
4	70.00000	NORMAL	MEAN= 70.00000, ST.DEV= 11.66666
5	39.00000	NORMAL	MEAN= 39.00000, ST.DEV= 4.59619
6	32.00000	BETA	MIN= 12.30770, MAX= 61.53848, ALPHA= 0.79428, BETA= 1.69143
7	13.00000	NORMAL	MEAN= 13.00000, ST.DEV= 2.14967
8	34.00000	NORMAL	MEAN= 34.00000, ST.DEV= 3.13846
9	5.00000	FIXED(CONSTANT)	TIME= 5.00000
10	21.00000	BETA	MIN= 9.54546, MAX= 38.18184, ALPHA= 12.99998, BETA= 20.00002
11	10.00000	FIXED(CONSTANT)	TIME= 10.00000
12	10.00000	NORMAL	MEAN= 10.00000, ST.DEV= 2.79508
13	44.00000	NORMAL	MEAN= 44.00000, ST.DEV= 9.83869
14	40.00000	NORMAL	MEAN= 40.00000, ST.DEV= 5.71428
15	80.00000	NORMAL	MEAN= 80.00000, ST.DEV= 13.71428
16	4.00000	FIXED(CONSTANT)	TIME= 4.00000
17	4.00000	BETA	MIN= 1.33333, MAX= 8.00000, ALPHA= 1.00000, BETA= 2.00000
18	6.00000	NORMAL	MEAN= 6.00000, ST.DEV= 1.69705

```
**********************************************************************************************************************
**********************************************************************************************************************
THIS IS THE CUTPUT FRCM THE MCDIFIED SIMPLIFICATION PRCGRAM: MODSIMP

**********************************************************************************************************************
**********************************************************************************************************************


     ACTIVITY      CRIGIN      TERMINAL      MEAN      STANDARD DEVIATICN
        2             1            2         20.8             5.13

                           F(T)    --    T
                           0.0     --    10.0000
                           0.05    --    12.7844
                           0.15    --    15.2036
                           0.25    --    16.6132
                           0.35    --    17.8740
                           0.45    --    18.9984
                           0.55    --    22.7115
                           0.65    --    22.7115
                           0.75    --    25.2538
                           0.85    --    25.2538
                           0.95    --    30.3384
                           1.00    --    32.8808


     ACTIVITY      ORIGIN      TERMINAL      MEAN      STANDARD DEVIATION
        4             2            6         70.0            10.94

                           F(T)          T
                           0.0     --    35.0000
                           0.05    --    50.8092
                           0.15    --    57.9081
                           0.25    --    62.1310
                           0.35    --    65.5046
                           0.45    --    68.5340
                           0.55    --    71.4660
                           0.65    --    74.4954
                           0.75    --    77.8690
                           0.85    --    82.0919
                           0.95    --    89.1908
                           1.00    --   105.0000
```

ACTIVITY	CRIGIN	TERMINAL	MEAN	STANDARD CEVIATION
5	2	5	39.0	4.31

F(T)	T
0.0 --	25.2114
0.05 --	31.4396
0.15 --	34.2363
0.25 --	35.8999
0.35 --	37.2290
0.45 --	38.4224
0.55 --	39.5776
0.65 --	40.7710
0.75 --	42.1001
0.85 --	43.7637
0.95 --	46.5604
1.00 --	52.7886

ACTIVITY	CRIGIN	TERMINAL	MEAN	STANDARD DEVIATION
6	2	4	31.9	10.03

F(T)	T
0.0 --	12.3077
0.05 --	16.5614
0.15 --	20.5854
0.25 --	23.8162
0.35 --	26.8016
0.45 --	29.7300
0.55 --	32.7243
0.65 --	35.9089
0.75 --	39.4651
0.85 --	43.7664
0.95 --	50.1376
1.00 --	61.5385

ACTIVITY	CRIGIN	TERMINAL	MEAN	STANDARD CEVIATION
7	2	3	13.0	2.02

F(T)	T
0.0 --	6.5510
0.05 --	9.4639
0.15 --	10.7720
0.25 --	11.5501
0.35 --	12.1717
0.45 --	12.7299
0.55 --	13.2701
0.65 --	13.8283
0.75 --	14.4499
0.85 --	15.2280
0.95 --	16.5361
1.00 --	19.4490

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
8	5	6	34.0	2.94

F(T)	T
0.0 --	24.5846
0.05 --	28.8375
0.15 --	30.7471
0.25 --	31.8831
0.35 --	32.7907
0.45 --	33.6056
0.55 --	34.3944
0.65 --	35.2093
0.75 --	36.1169
0.85 --	37.2529
0.95 --	39.1625
1.00 --	43.4154

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
9	4	5	5.0	0.00

F(T)	T
0.0 --	5.0000
0.05 --	5.0000
0.15 --	5.0000
0.25 --	5.0000
0.35 --	5.0000
0.45 --	5.0000
0.55 --	5.0000
0.65 --	5.0000
0.75 --	5.0000
0.85 --	5.0000
0.95 --	5.0000
1.00 --	5.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
10	3	4	21.0	2.21

F(T)	T
0.0 --	9.5455
0.05 --	17.2433
0.15 --	18.5502
0.25 --	19.3660
0.35 --	20.0346
0.45 --	20.6459
0.55 --	21.2456
0.65 --	21.8719
0.75 --	22.5750
0.85 --	23.4600
0.95 --	24.9458
1.00 --	38.1818

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
11	6	8	10.0	0.00

F(T)	T
0.0 --	10.0000
0.05 --	10.0000
0.15 --	10.0000
0.25 --	10.0000
0.35 --	10.0000
0.45 --	10.0000
0.55 --	10.0000
0.65 --	10.0000
0.75 --	10.0000
0.85 --	10.0000
0.95 --	10.0000
1.00 --	10.0000

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
12	6	7	10.0	2.62

F(T)	T
0.0 --	1.6147
0.05 --	5.4023
0.15 --	7.1030
0.25 --	8.1147
0.35 --	8.9230
0.45 --	9.6488
0.55 --	10.3512
0.65 --	11.0770
0.75 --	11.8853
0.85 --	12.8970
0.95 --	14.5977
1.00 --	18.3853

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
13	5	7	44.0	9.23

F(T)	T
0.0 --	14.4839
0.05 --	27.8160
0.15 --	33.8027
0.25 --	37.3639
0.35 --	40.2090
0.45 --	42.7637
0.55 --	45.2363
0.65 --	47.7910
0.75 --	50.6361
0.85 --	54.1973
0.95 --	60.1840
1.00 --	73.5161

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
14	4	7	40.0	5.36

F(T)	T
0.0 --	22.8572
0.05 --	30.6004
0.15 --	34.0774
0.25 --	36.1458
0.35 --	37.7982
0.45 --	39.2819
0.55 --	40.7181
0.65 --	42.2018
0.75 --	43.8542
0.85 --	45.9226
0.95 --	49.3996
1.00 --	57.1428

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
15	3	9	80.0	12.86

F(T)	T
0.0 --	38.8572
0.05 --	57.4410
0.15 --	65.7858
0.25 --	70.7499
0.35 --	74.7156
0.45 --	78.2767
0.55 --	81.7233
0.65 --	85.2844
0.75 --	89.2501
0.85 --	94.2142
0.95 --	102.5590
1.00 --	121.1428

ACTIVITY	ORIGIN	TERMINAL	MEAN	STANDARD DEVIATION
16	7	8	4.0	0.00

F(T)	T
0.0 --	4.0000
0.05 --	4.0000
0.15 --	4.0000
0.25 --	4.0000
0.35 --	4.0000
0.45 --	4.0000
0.55 --	4.0000
0.65 --	4.0000
0.75 --	4.0000
0.85 --	4.0000
0.95 --	4.0000
1.00 --	4.0000

ACTIVITY	CRIGIN	TERMINAL	MEAN	STANDARD CEVIATION
17	7	9	4.0	1.30

F(T)	T
0.0 --	1.3333
0.05 --	1.9840
0.15 --	2.5292
0.25 --	2.9535
0.35 --	3.3398
0.45 --	3.7155
0.55 --	4.0979
0.65 --	4.5038
0.75 --	4.9579
0.85 --	5.5103
0.95 --	6.3427
1.00 --	8.0000

ACTIVITY	CRIGIN	TERMINAL	MEAN	STANDARD CEVIATION
18	8	9	6.0	1.59

F(T)	T
0.0 --	0.9088
0.05 --	3.2085
0.15 --	4.2411
0.25 --	4.8554
0.35 --	5.3461
0.45 --	5.7867
0.55 --	6.2133
0.65 --	6.6539
0.75 --	7.1446
0.85 --	7.7589
0.95 --	8.7915
1.00 --	11.0912

```
****************************************************************************************************
****************************************************************************************************

THIS IS THE OUTPUT FROM THE SUBNETWORK ANALYSIS PROGRAM: SUBNET

****************************************************************************************************
****************************************************************************************************


EACH SUBNETWORK'S APPROXIMATE CURATION DISTRIBUTION IS DETERMINED.
```

SUBNETWORK 2

INITIAL INPUT

ACTIVITY	ORIGIN	TERMINAL	LOWER POINT	UPPER POINT	MEAN	STANDARD DEVIATION	PROB. LOWER PT.
1	1	2	16.1595	26.4703	20.7743	5.1270	0.55243

THE DURATION DISTRIBUTION FOR SUBNETWORK 2

TIME	DISTRIBUTION
0.10000D 02	0.0
0.12784D 02	0.50000D-01
0.15204D 02	0.15000D 00
0.16613D 02	0.25000D 00
0.17874D 02	0.35000D 00
0.18558D 02	0.45000D 00
0.22712D 02	0.55000D 00
0.22712D 02	0.65000D 00
0.25254D 02	0.75000D 00
0.25254D 02	0.85000D 00
0.30338D 02	0.95000D 00
0.32881D 02	0.10000D 01

SUBNETWORK 3

INITIAL INPUT

ACTIVITY	ORIGIN	TERMINAL	LOWER POINT	UPPER POINT	MEAN	STANDARD DEVIATION	PROB. LOWER PT.
1	1	5	59.0567	80.9433	70.0000	10.9433	0.50000
2	1	4	34.6888	43.3112	39.0000	4.3112	0.50000
3	1	3	23.0185	43.2217	31.9497	10.0336	0.55793
4	1	2	10.9836	15.0164	13.0000	2.0164	0.50000
5	5	7	10.0000	10.0000	10.0000	0.0	1.00000
6	5	6	7.3782	12.6218	10.0000	2.6218	0.50000
7	4	5	31.0561	36.9439	34.0000	2.9439	0.50000
8	4	6	34.7713	53.2287	44.0000	9.2287	0.50000
9	3	4	5.0000	5.0000	5.0000	0.0	1.00000
10	3	6	34.6400	45.3600	40.0000	5.3600	0.50000
11	2	3	18.8738	23.2939	20.9938	2.2082	0.52037
12	2	8	67.1360	92.8640	80.0000	12.8640	0.50000
13	7	8	4.4082	7.5918	6.0000	1.5918	0.50000
14	6	7	4.0000	4.0000	4.0000	0.0	1.00000
15	6	8	2.6330	5.4402	3.9934	1.2957	0.55491

```
THE ORDER IN WHICH TO CONSIDER THE ACTIVITIES TO DETERMINE THE LONGEST PATH TIME:

     ORDER  ACTIVITY
        1         1
        2         2
        3         3
        4         4
        5        11
        6        12
        7         9
        8        10
        9         7
       10         8
       11         5
       12         6
       13        14
       14        15
       15        13

THE CRITICAL PATH TIME WHEN EACH ACTIVITY'S COMPLETION TIME IS SET EQUAL TO ITS MEAN IS =     0.93000D 02

     THE   3 NODES ON THE CRITICAL PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
           8.  2.  1.

     THE   2 CRITICAL ACTIVITIES ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
          12.  4.

A LOWER BOUND ON THE SUBNETWORK DURATION TIME IS =     0.81700D 02

     THE   8 NODES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
           8.  7.  6.  5.  4.  3.  2.  1.

     THE   7 ACTIVITIES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
          13. 14.  6.  7.  9. 11.  4.

A UPPER BOUND ON THE SUBNETWORK DURATION TIME IS =     0.11304D 03

     THE   6 NODES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL NODE:
           8.  7.  6.  4.  3.  1.

     THE   5 ACTIVITIES ON THE LONGEST PATH ARE AS FOLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:
          13. 14.  8.  9.  3.
```

AD-A060 568 TEXAS A AND M UNIV COLLEGE STATION INST OF STATISTICS F/G 9/2
A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJEC--ETC(U)
AUG 78 T C BAKER, R L SIELKEN N00014-78-C-0426

UNCLASSIFIED THEMIS-TR-57 NL

2 OF 4
AD A060568

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.100000 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  12. IS =    5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13. 14.  6.  7.  2.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   4. IS =    5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13. 14.  6.  7.  2.

THERE ARE    1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE    1-TH CLUSTER ARE AS FOLLOWS:
          12. 13. 14.  6.  7.  2.  4.

     THERE ARE    8 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =     0.10000D 01

     THERE ARE  13 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
             1.        1
             2.        2
             3.        3
             4.        5
             5.        6
             6.        7
             7.        8
             8.        9
             9.       10
            10.       11
            11.       13
            12.       14
            13.       15

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    1
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    2
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    3
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    6
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    7
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    8
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    9

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   10

     THERE ARE    1 ELIMINANTS CORRESPONDING TO ACTIVITY   11
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY   11 IS ACTIVITY  12

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY   13
         THE   1-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY  12
         THE   2-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   14

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   15
```

```
THERE ARE    1 NCNEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

     THERE ARE  11 ACTIVITIES IN THE   1-TH CLUSTER.  THEY ARE AS FOLLOWS:
               12. 13. 14.  6.  7.  2.  4.  1.  3.  8. 11.

          2 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FOLLOWS:
               1.  2.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA,LAMBDA) = (0.10000D 01 , 0.10000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.83979D 02	0.23224D-02	0.23224D-02
0.86259D 02	0.23224D-02	0.23224D-02
0.88538D 02	0.46447D-02	0.46447D-02
0.90818D 02	0.13934D-01	0.18215D-01
0.93097D 02	0.22678D-01	0.22678D-01
0.95377D 02	0.36067D-01	0.43139D-01
0.97656D 02	0.90297D-01	0.90297D-01
0.99935D 02	0.12941D 00	0.13479D 00
0.10221D 03	0.20336D 00	0.20336D 00
0.10449D 03	0.38347D 00	0.38347D 00
0.10677D 03	0.50814D 00	0.50814D 00
0.10747D 03	0.50814D 00	0.50814D 00
0.10817D 03	0.75049D 00	0.75049D 00
0.10886D 03	0.75049D 00	0.75049D 00
0.10956D 03	0.77906D 00	0.77906D 00
0.11026D 03	0.90395D 00	0.90395D 00
0.11095D 03	0.90395D 00	0.90395D 00
0.11165D 03	0.90395D 00	0.90395D 00
0.11235D 03	0.90395D 00	0.90395D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.20000D 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY. I.E. ACTIVITY  12. IS =   5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13. 14.  6.  7.  2.

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY. I.E. ACTIVITY   4. IS =   5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13. 14.  6.  7.  2.

THERE ARE    1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE    1-TH CLUSTER ARE AS FOLLOWS:
          12. 13. 14.  6.  7.  2.  4.

     THERE ARE    8 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =      0.20000D 01

     THERE ARE  13 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
             1.       1
             2.       2
             3.       3
             4.       5
             5.       6
             6.       7
             7.       8
             8.       9
             9.      10
            10.      11
            11.      13
            12.      14
            13.      15

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    1
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    2
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    3
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    6
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    7
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    8
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    9

     THERE ARE    1 ELIMINANTS CORRESPONDING TO ACTIVITY   10
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   10 IS ACTIVITY  12

     THERE ARE    1 ELIMINANTS CORRESPONDING TO ACTIVITY   11
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   11 IS ACTIVITY  12

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY   13
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   14

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   15
```

```
THERE ARE   1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

     THERE ARE  12 ACTIVITIES IN THE   1-TH CLUSTER.  THEY ARE AS FOLLOWS:
               12. 13. 14.  6.  7.  2.  4.  1.  3.  8. 10. 11.

          2 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FOLLOWS:
               1.  2.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA.LAMBDA) = (0.20000D 01 . 0.20000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.83979D 02	0.23178D-02	0.23178D-02
0.86259D 02	0.23178D-02	0.23178D-02
0.88538D 02	0.46355D-02	0.46355D-02
0.90818D 02	0.20316D-01	0.20316D-01
0.93097D 02	0.22633D-01	0.22633D-01
0.95377D 02	0.35214D-01	0.35214D-01
0.97656D 02	0.91031D-01	0.91031D-01
0.99935D 02	0.13791D 00	0.13791D 00
0.10221D 03	0.21670D 00	0.21670D 00
0.10449D 03	0.37580D 00	0.37580D 00
0.10677D 03	0.51085D 00	0.51085D 00
0.10747D 03	0.51085D 00	0.51085D 00
0.10817D 03	0.76481D 00	0.76481D 00
0.10886D 03	0.76481D 00	0.76481D 00
0.10956D 03	0.78091D 00	0.78091D 00
0.11026D 03	0.90824D 00	0.90824D 00
0.11095D 03	0.90824D 00	0.90824D 00
0.11165D 03	0.90824D 00	0.90824D 00
0.11235D 03	0.90824D 00	0.90824D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

```
THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA =      0.20000D 01

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    1-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY  12, IS =   5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13, 14,  6,  7,  2,

     THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE    2-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY   4, IS =   5

          THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLOWS
          13, 14,  6,  7,  2,

THERE ARE    1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF ASSOCIATES ONLY.

     THE ACTIVITIES IN THE   1-TH CLUSTER ARE AS FOLLOWS:
          12, 13, 14,  6,  7,  2,  4,

     THERE ARE    8 ACTIVITIES NOT IN ANY CLUSTER YET.
```

```
THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY ARE NOW DETERMINED FOR THETA =      0.30000D 01

     THERE ARE  13 ACTIVITIES NOT ON THE CRITICAL PATH.  THEY ARE AS FOLLOWS:
           1.      1
           2.      2
           3.      3
           4.      5
           5.      6
           6.      7
           7.      8
           8.      9
           9.     10
          10.     11
          11.     13
          12.     14
          13.     15

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    1
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    1 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    2
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    2 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    3
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    3 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    5

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    6
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    6 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    7
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    7 IS ACTIVITY   4

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY    8
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY    8 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY    9

     THERE ARE    1 ELIMINANTS CORRESPONDING TO ACTIVITY   10
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   10 IS ACTIVITY  12

     THERE ARE    1 ELIMINANTS CORRESPONDING TO ACTIVITY   11
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   11 IS ACTIVITY  12

     THERE ARE    2 ELIMINANTS CORRESPONDING TO ACTIVITY   13
         THE    1-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY  12
         THE    2-TH ELIMINANT CORRESPONDING TO ACTIVITY   13 IS ACTIVITY   4

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   14

     THERE ARE    0 ELIMINANTS CORRESPONDING TO ACTIVITY   15
```

```
THERE ARE   1 NONEMPTY CLUSTERS AFTER POOLING ON THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.

    THERE ARE  12 ACTIVITIES IN THE   1-TH CLUSTER.  THEY ARE AS FOLLOWS:
          12. 13. 14.  6.  7.  2.  4.  1.  3.  8. 10. 11.

      2 CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER. THEY WERE AS FOLLOWS:
           1.  2.
```

BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK 3 WHEN (THETA,LAMBDA) = (0.30000D 01 , 0.20000D 01)

TIME	LOWER BOUND	UPPER BOUND
0.83979D 02	0.23178D-02	0.23178D-02
0.86259D 02	0.23178D-02	0.23178D-02
0.88538D 02	0.46355D-02	0.46355D-02
0.90818D 02	0.20316D-01	0.20316D-01
0.93097D 02	0.22633D-01	0.22633D-01
0.95377D 02	0.35214D-01	0.35214D-01
0.97656D 02	0.91031D-01	0.91031D-01
0.99935D 02	0.13791D 00	0.13791D 00
0.10221D 03	0.21670D 00	0.21670D 00
0.10449D 03	0.37580D 00	0.37580D 00
0.10677D 03	0.51085D 00	0.51085D 00
0.10747D 03	0.51085D 00	0.51085D 00
0.10817D 03	0.76481D 00	0.76481D 00
0.10886D 03	0.76481D 00	0.76481D 00
0.10956D 03	0.78091D 00	0.78091D 00
0.11026D 03	0.90824D 00	0.90824D 00
0.11095D 03	0.90824D 00	0.90824D 00
0.11165D 03	0.90824D 00	0.90824D 00
0.11235D 03	0.90824D 00	0.90824D 00
0.11304D 03	0.10000D 01	0.10000D 01

THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTER PROCEDURE.

ONLY 500 OF THE POSSIBLE ACTIVITY DURATION CONFIGURATIONS WERE CONSIDERED.

SYSTEMATIC SAMPLING WAS USED.

THE UPPER AND LOWER BOUNDS ON THE DISCRETE SUBNETWORK DURATION DISTRIBUTION ARE EQUAL.

THUS AN ESTIMATE OF THE DISCRETE DURATION DISTRIBUTION FOR SUBNETWORK 3 IS

TIME	DISTRIBUTION
0.83979D 02	0.23178D-02
0.86259D 02	0.23178D-02
0.88538D 02	0.46355D-02
0.90818D 02	0.20316D-01
0.93097D 02	0.22633D-01
0.95377D 02	0.35214D-01
0.97656D 02	0.91031D-01
0.99935D 02	0.13791D 00
0.10221D 03	0.21670D 00
0.10449D 03	0.37580D 00
0.10677D 03	0.51085D 00
0.10747D 03	0.51085D 00
0.10817D 03	0.76481D 00
0.10886D 03	0.76481D 00
0.10956D 03	0.78091D 00
0.11026D 03	0.90824D 00
0.11095D 03	0.90824D 00
0.11165D 03	0.90824D 00
0.11235D 03	0.90824D 00
0.11304D 03	0.10000D 01

```
****************************************************************************************************
****************************************************************************************************

THIS IS THE OUTPUT FROM THE SYNTHESIS PROGRAM: SYNTH

****************************************************************************************************
****************************************************************************************************



THE CURRENT PROJECT SCHEDULE IS THE ONE MOST RECENTLY LISTED BY THE DETERMINISTIC SCHEDULE RESOLUTION PROGRAM.

THE PROJECT'S CORRESPONDING APPROXIMATE COMPLETION TIME DISTRIBUTION IS DETERMINED BELOW:


FOR THIS PROBLEM

THE NUMBER OF C.D.F. SUBDIVISIONS USED THROUGHOUT SYNTHESIS IS   49.
THE TARGET TIME USED BY THE DETERMINISTIC SCHEDULER WAS  112.07396
THE MEAN OF THE PROJECT'S APPROXIMATE COMPLETION TIME DISTRIBUTION WILL BE COMPARED TO THE
SPECIFIED PROJECT DEADLINE TIME OF     125.00000


     THE INSTRUCTIONS TO BE PERFORMED ARE


                                       SERIES=0
   INSTRUCTION NO.     SUBNETWORK      PARALLEL=1      SUBNETWORKS TO BE SYNTHESIZED

          1                 1               0              2  3
```

SUBNETWORK	APPROXIMATE DISTRIBUTION DETERMINED USING SUBNETWORK ANALYSIS	
2	X	F(X)
	10.0000	0.0
	12.7843	0.05000
	15.2036	0.15000
	16.6132	0.25000
	17.8740	0.35000
	18.9984	0.45000
	22.7115	0.55000
	22.7115	0.65000
	25.2538	0.75000
	25.2538	0.85000
	30.3384	0.95000
	32.8807	1.00000
3	X	F(X)
	83.9793	0.00232
	86.2587	0.00232
	88.5382	0.00464
	90.8176	0.02032
	93.0971	0.02263
	95.3765	0.03521
	97.6560	0.09103
	99.9354	0.13791
	102.2148	0.21670
	104.4943	0.37580
	106.7737	0.51085
	107.4702	0.51085
	108.1667	0.76481
	108.8632	0.76481
	109.5597	0.78091
	110.2562	0.90824
	110.9527	0.90824
	111.6492	0.90824
	112.3457	0.90824
	113.0422	1.00000

THE DISTRIBUTION FOR THE SYNTHESIZED NETWORK IS

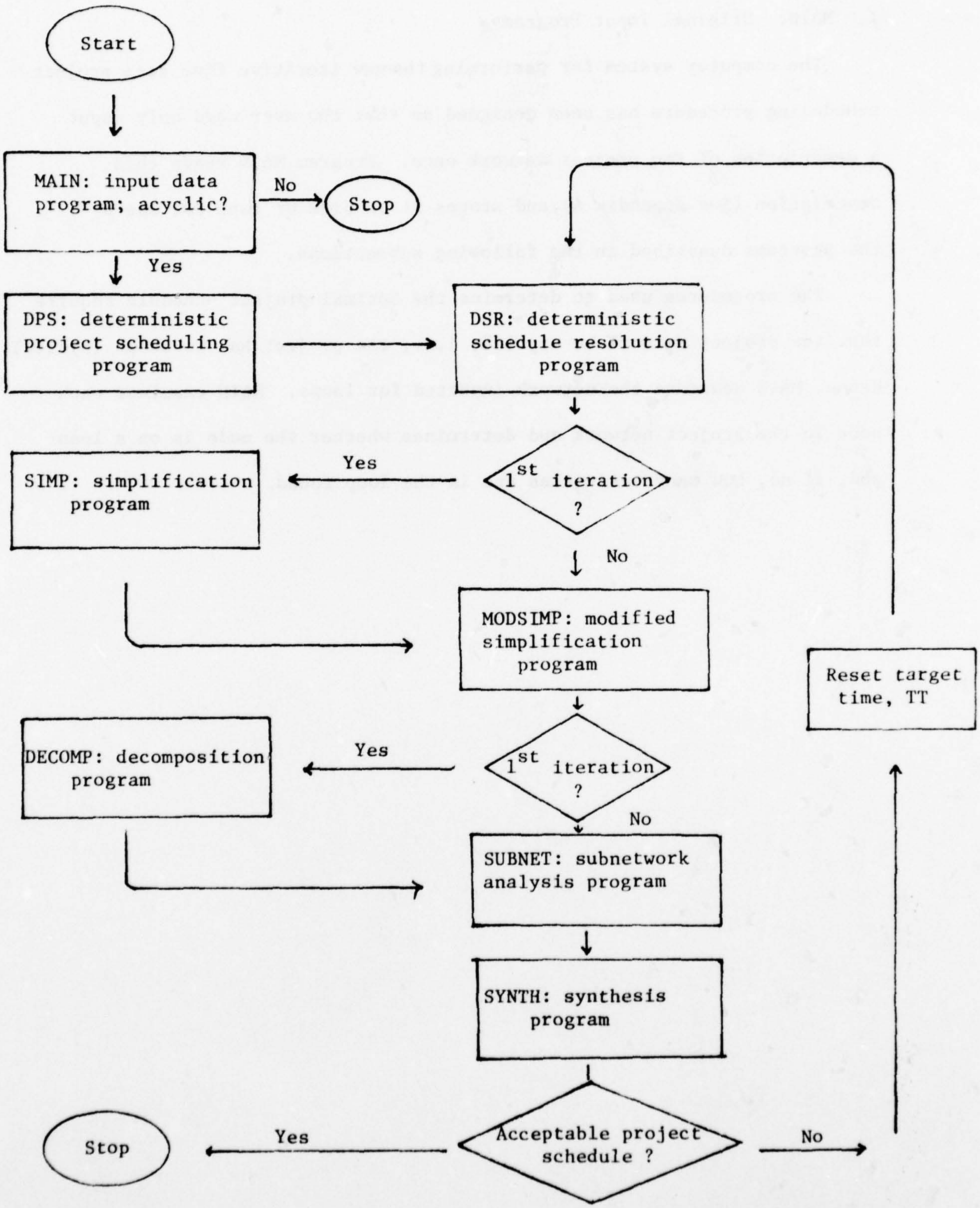
T	F(T)
93.97934	0.0
95.03941	0.0
96.09949	0.0
97.15956	0.00012
98.21964	0.00012
99.27971	0.00035
100.33979	0.00035
101.39986	0.00070
102.45993	0.00093
103.52001	0.00116
104.58008	0.00217
105.64016	0.00241
106.70023	0.00479
107.76031	0.00659
108.82038	0.00901
109.88046	0.01128
110.94053	0.01533
112.00061	0.01728
113.06068	0.02544
114.12076	0.03030
115.18083	0.04599
116.24090	0.05518
117.30098	0.07352
118.36105	0.08906
119.42113	0.10186
120.48120	0.14357
121.54128	0.18425
122.60135	0.21447
123.66143	0.29642
124.72150	0.31071
125.78158	0.39035
126.84165	0.42206
127.90172	0.50937
128.96180	0.53910
130.02187	0.61984
131.08195	0.68729
132.14202	0.72347
133.20210	0.76238
134.26217	0.81318
135.32225	0.83625
136.38232	0.88006
137.44240	0.90152
138.50247	0.91987
139.56255	0.94527
140.62262	0.96637
141.68269	0.97906
142.74277	0.97987
143.80284	0.99541
144.86292	0.99541
145.92299	1.00000

THE MEAN OF THE SYNTHESIZED DISTRIBUTION IS 128.42171

THE DIFFERENCE BETWEEN 128.42171 AND THE PROJECT DEADLINE IS 2.74 PERCENT OF THE PROJECT DEADLINE.
HENCE. THE NEW TARGET TIME IS 109.08782

```
*******************************************************************************************************************
THIS COMPLETES ITERATION     2
*******************************************************************************************************************
```

Appendix D. Flowchart of the Computer System



Appendix E

Program Descriptions

1. MAIN: Original Input Program

The computer system for performing the new iterative five step project scheduling procedure has been designed so that the user need only input a description of the project network once. Program MAIN reads this description (See Appendix A) and stores it on disk or tape for use by the programs described in the following subsections.

The procedures used to determine the optimal project schedule require that the project network be acyclic; i.e., the project has no loops (cycles). Hence, MAIN searches the network inputted for loops. MAIN examines each node in the project network and determines whether the node is on a loop and, if so, how many activities are in the loop found.

2. DPS: Deterministic Project Scheduling

The problem of finding a minimum cost project schedule which completes the project by TARGET TIME when each activity's duration is exactly its mean duration can be formulated as a linear programming problem. However, due to the large number of variables and constraints involved, a straightforward linear programming solution would be impractical. Instead the dual of this linear programming problem is considered, further reformulated, and then solved using the very efficient network-flow algorithm described in Dunn and Sielken (1977). This network-flow algorithm is a generalization of D. R. Fulkerson's algorithm (1961) for solving similar problems with linear activity cost functions. The generalized network-flow algorithm iteratively generates the minimum cost project schedule for every feasible deterministic completion deadline. The corresponding deterministic project cost curve is a convex piecewise linear function of TARGET TIME and a valuable description of the relationship between a project's cost and its deadline. The optimal activity mean durations are linear functions of the TARGET TIME on each linear piece of the project cost curve.

3. DSR: Deterministic Schedule Resolution

Since the deterministic project scheduler iteratively generates the optimal activity mean durations for all feasible completion deadlines, Step 1 is essentially only performed once. When the general iterative algorithm returns to Step 1 with a new TARGET TIME, finding the optimal activity mean durations is essentially a simple table look-up procedure. For a more complete detailed discussion of Deterministic Scheduling see Dunn and Sielken (1977).

4. SIMP: Simplification

Five configurations of activities for which a single equivalent activity and duration distribution are readily available are depicted in Figure E.1. The equivalent single activity duration distributions for the parallel, series, and Wheatstone Bridge configurations were originally identified by Hartley and Wortham (1966) and for the Double Wheatstone Bridge and Criss-Cross configurations by Ringer (1969).

Simplification is an iterative procedure as illustrated in Figure E.2. In the special case where Simplification reduces the project network down to just one activity as in Figure E.2, the project completion time distribution is directly determined so that Steps 3 and 4 are skipped. Although a reduction to one activity is a very special case, reductions of over 50% are quite common.

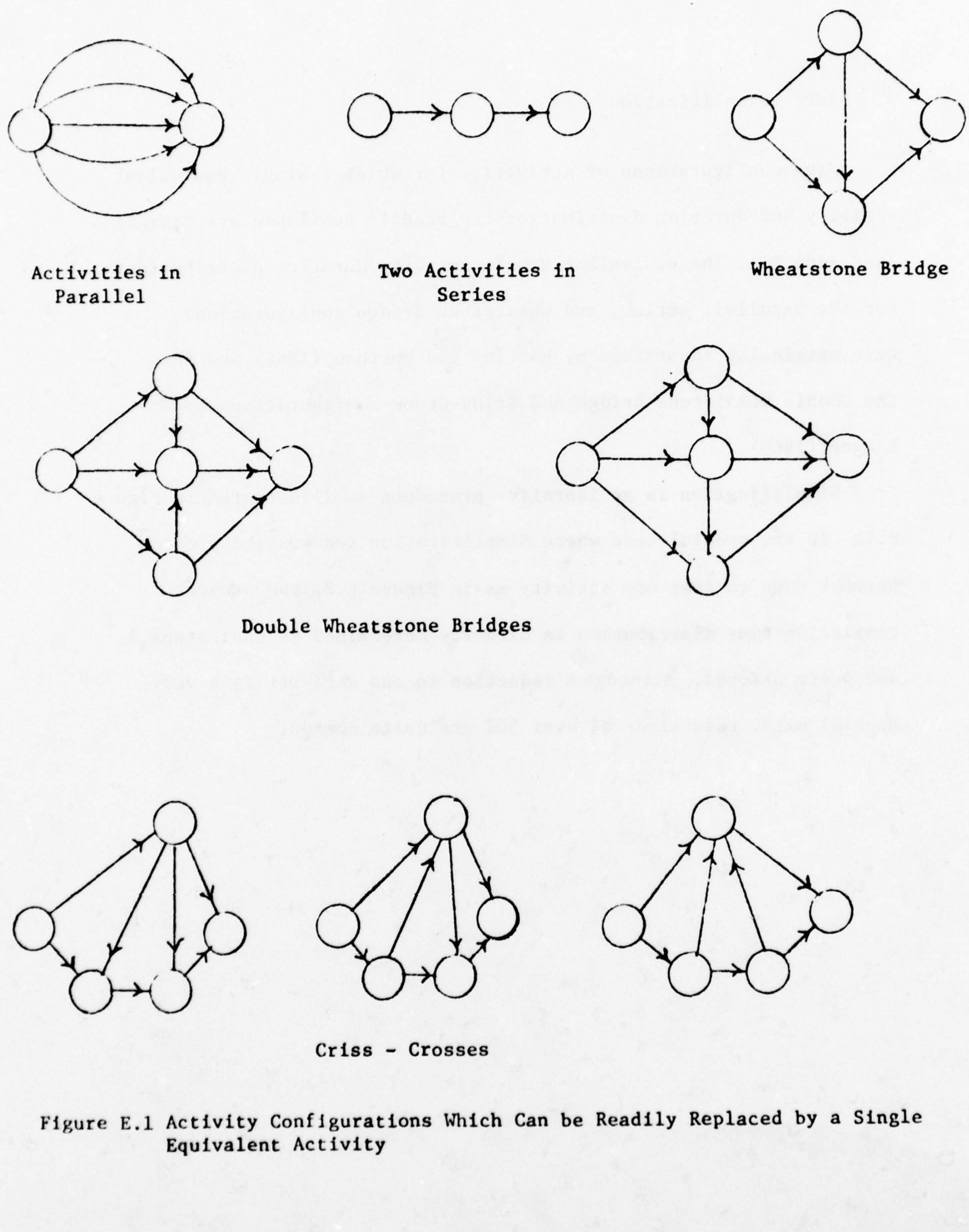
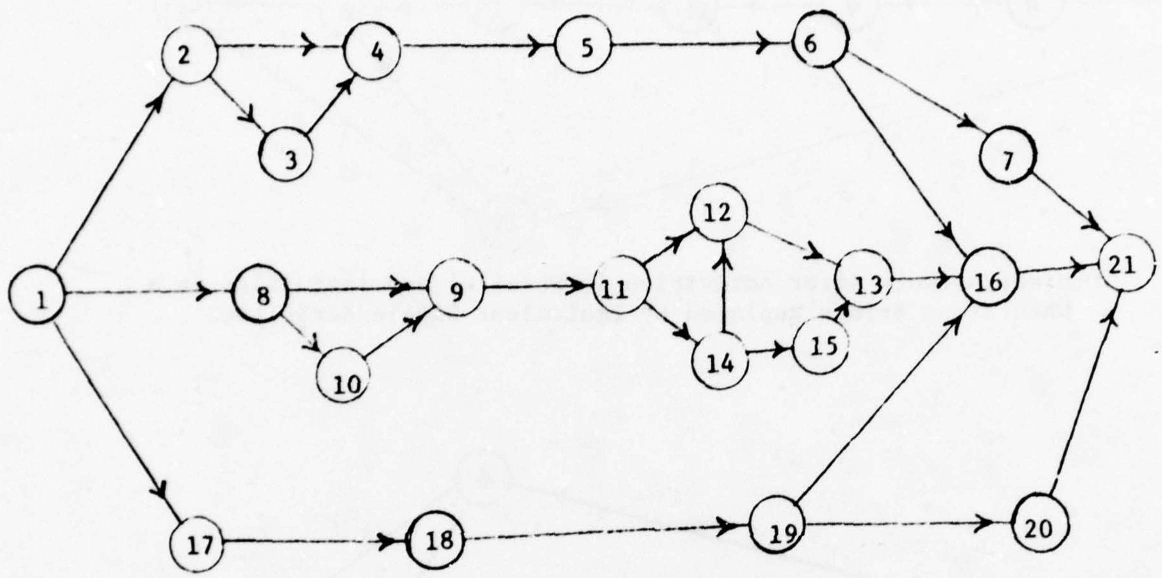
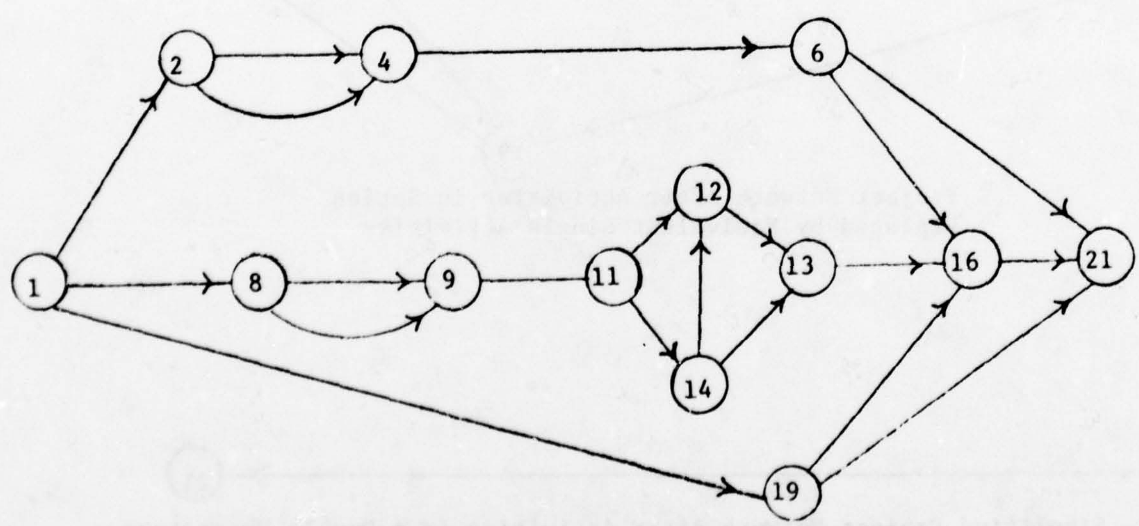


Figure E.1 Activity Configurations Which Can be Readily Replaced by a Single Equivalent Activity

Figure E.2 The Iterative Simplification of a Project Network

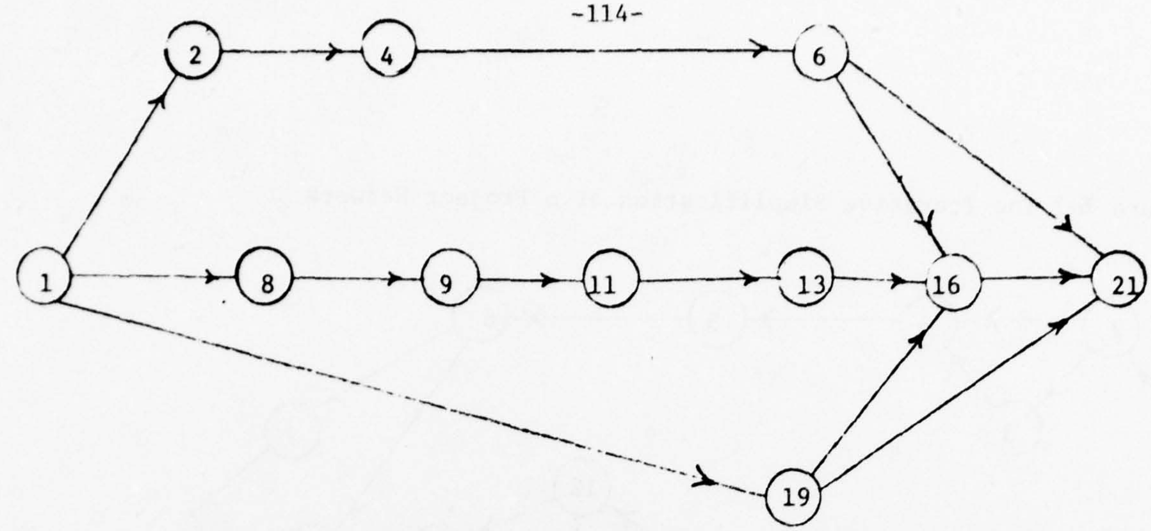


Original Project Network

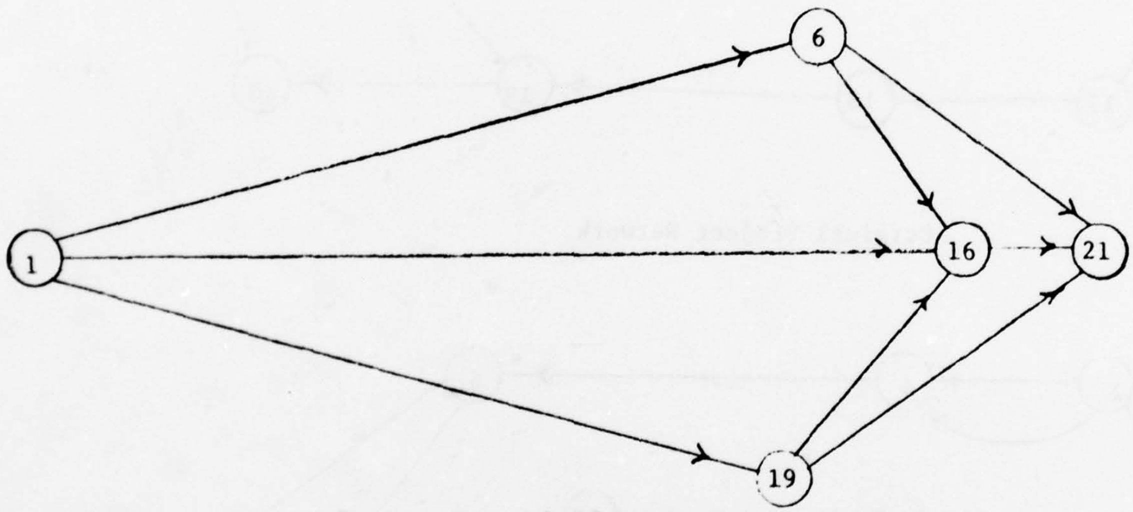


Project Network After Activities in Series
Replaced by Equivalent Single Activities

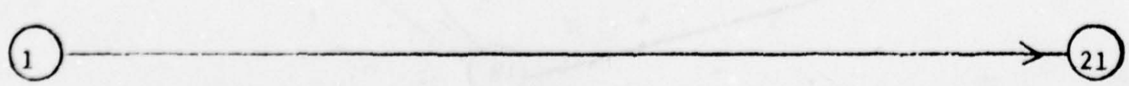
Figure E.2(Continued)



Project Network After Activities in Parallel and Activities in a Wheatstone Bridge Replaced by Equivalent Single Activities



Project Network After Activities in Series Replaced by Equivalent Single Activities



Simplified Project Network After Activities in a Double Wheatstone Bridge Replaced by an Equivalent Single Activity

5. MODSIMP: Subsequent Simplification

The structure of the simplified network is identified by SIMP. MODSIMP determines the distribution of the activity durations in the simplified network.

6. SUBNET: Subnetwork Analysis

6.1 Introduction

The objective of Subnetwork Analysis is to determine each subnetwork's duration distribution.

At the end of Step 2 each activity in the subnetwork has a specified duration distribution. This distribution is now approximated by a two-point discrete distribution. In particular, an activity, say A, is now conceptualized as having two possible duration times, say ℓ_A for a lower duration and u_A for an upper duration. The probability that the activity duration is ℓ_A is assumed to be P_A, and correspondingly the probability that the activity duration is u_A is assumed to be $Q_A = 1 - P_A$. The values of ℓ_A, u_A, and P_A are chosen so that the mean, variance, and third moment of the discrete distribution are the same as the mean, variance, and third moment of activity A's specified duration distribution.

Let n be the number of activities in the subnetwork. Let $v = 1, 2, \ldots, 2^n$ index the 2^n possible configurations of activity durations when each activity is either at its upper duration or at its lower duration. Let

$$p_v = \text{probability of the v-th activity duration configuration}$$

$$= \prod_{i=1}^{n} [P_i(1 - \delta_{v,i}) + Q_i\delta_{v,i}] \qquad (6.1)$$

where

$$\delta_{v,i} = 1 \text{ if the duration for the i-th activity is } u_i \text{ in the v-th activity duration configuration}$$
$$= 0 \text{ if the duration for the i-th activity is } \ell_i \text{ in the v-th activity duration configuration.} \quad (6.2)$$

Then the subnetwork duration distribution when each activity has its two-point discrete distribution is

$$F(t) = \sum_{v=1}^{2^n} p_v I_t(t_v) \quad (6.3)$$

where

t_v = the subnetwork duration when the activity durations are in the v-th configuration (6.4)

and

$$I_t(t_v) = 1 \text{ if } t_v \leq t ,$$
$$= 0 \text{ if } t_v > t . \quad (6.5)$$

The discrete distribution function F is an approximation to the subnetwork's exact duration distribution.

The goal of Subnetwork Analysis is to determine F.

Since the number, n, of activities in the subnetwork may be fairly large, the complete enumeration of the 2^n discrete subnetwork durations may sometimes be impractical. When this happens, the discrete subnetwork duration distribution F must be approximated. The approximation of F will be based on the activities which are mostly likely to influence the subnetwork duration. The identification of these important activities and their interrelationships is discussed in the next subsection which is a review of the procedures originating in Sielken,

Ringer, Hartley, and Arseven (1974) and Sielken, Hartley, and Spoerl (1976).

Each subnetwork is assumed to be an acyclic network with one source, one sink, and no cut vertices.

6.2 Formation of Clusters

The mean duration for activity A is

$$m_A = P_A \ell_A + Q_A u_A , \tag{6.6}$$

and the standard deviation of activity A's duration is

$$s_A = [P_A \ell_A^2 + Q_A u_A^2 - m_A^2]^{\frac{1}{2}} . \tag{6.7}$$

When each activity duration takes on a fixed (nonrandom) value, the subnetwork's duration is the duration of the longest path through the subnetwork where the "length" of an activity is its duration. For example, consider the subnetwork described in Table E.1 and displayed in Figure E.3. When each activity duration is its mean duration, then the subnetwork's duration is 32, corresponding to the path consisting of activities 2, 7, and 9.

Definition 1: A critical activity is an activity on the longest path when all the subnetwork's activity durations are set to their means.

Thus in the example the critical activities are 2, 7, and 9.

The search for the activities which are most likely to influence the subnetwork duration begins with the critical activities. Each critical activity initiates a separate set of activities called a

TABLE E.1

Activity Durations for the Subnetwork in Figure 2

Activity	ℓ_A	u_A	P_A	m_A	s_A
1	0.00	2.00	.5	1	1
2	8.00	10.50	.2	10	1
3	9.55	15.67	.6	12	3
4	1.50	4.00	.8	2	1
5	3.35	5.52	.7	4	1
6	4.00	6.00	.5	5	1
7	8.73	16.90	.6	12	4
8	12.00	14.50	.2	14	1
9	5.00	15.00	.5	10	5

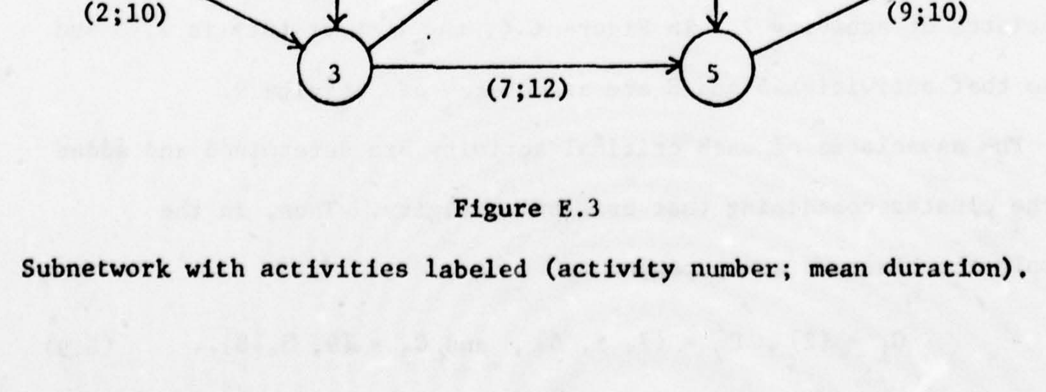


Figure E.3

Subnetwork with activities labeled (activity number; mean duration).

"cluster". Initially there are several clusters. In the example the initial clusters are

$$C_1 = \{2\}\ ,\ C_2 = \{7\}\ ,\ \text{and } C_3 = \{9\}\ . \qquad (6.8)$$

Some of the non-critical activities may influence the subnetwork's duration when not all of the activity durations are at their mean values.

<u>Definition 2</u>: An <u>associate</u> of a critical activity A is a non-critical activity which is on the longest path when all activity durations are set to their mean except for the critical activity A which has its duration reduced from m_A to $\max(m_A - \lambda s_A, 0)$ where λ is a nonnegative parameter.

Thus the associates of a critical activity A are those activities whose effect on the subnetwork's duration are related to activity A's duration. In the example, for $\lambda = 1$ the associates of the critical activities 2, 7, and 9 can be determined by considering Figures E.4, E.5, and E.6 respectively. In Figure E.4 the longest path is still the critical path 2, 7, and 9, so that activity 2 has no associates. In Figure E.5 the longest path is 2, 5, 6, and 9, so that activities 5 and 6 are the associates of activity 7. In Figure E.6, the longest path is 2, 5 and 8, so that activities 5 and 8 are associates of activity 9.

The associates of each critical activity are determined and added to the cluster containing that critical activity. Thus, in the example the clusters are expanded to

$$C_1 = \{2\}\ ,\ C_2 = \{7, 5, 6\}\ ,\ \text{and } C_3 = \{9, 5, 8\}\ . \qquad (6.9)$$

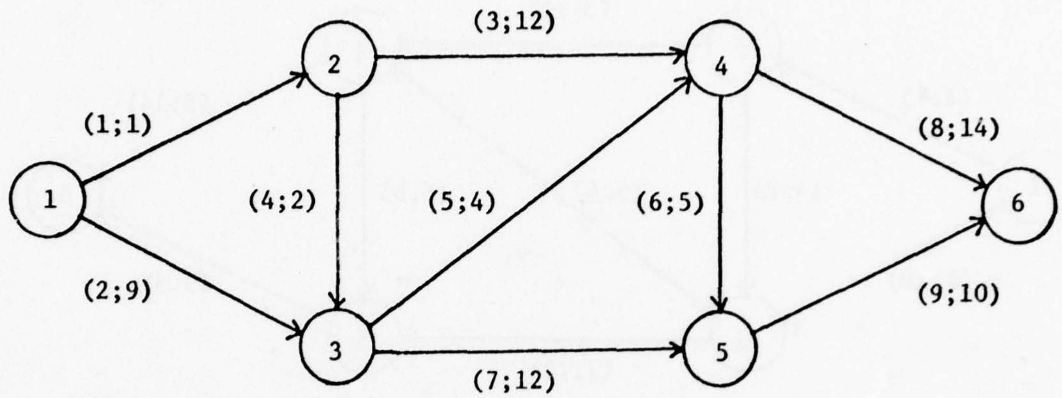


Figure E.4

Subnetwork for determining the associates of Activity 2 when $\lambda = 1$.

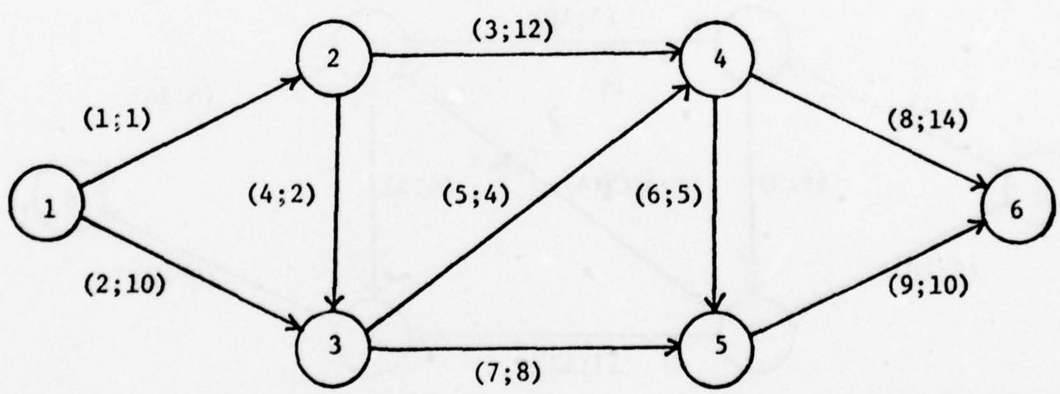


Figure E.5

Subnetwork for determining the associates of Activity 7 when $\lambda = 1$.

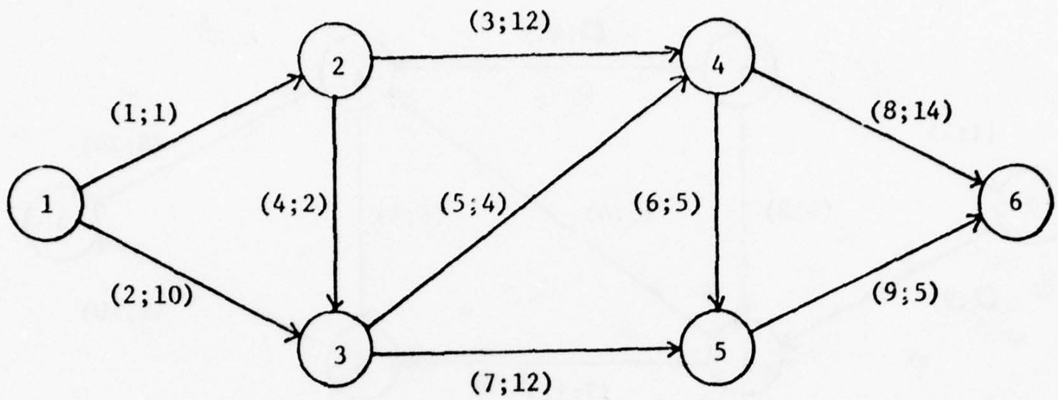


Figure E.6

Subnetwork for determining the associates of Activity 9 when $\lambda = 1$.

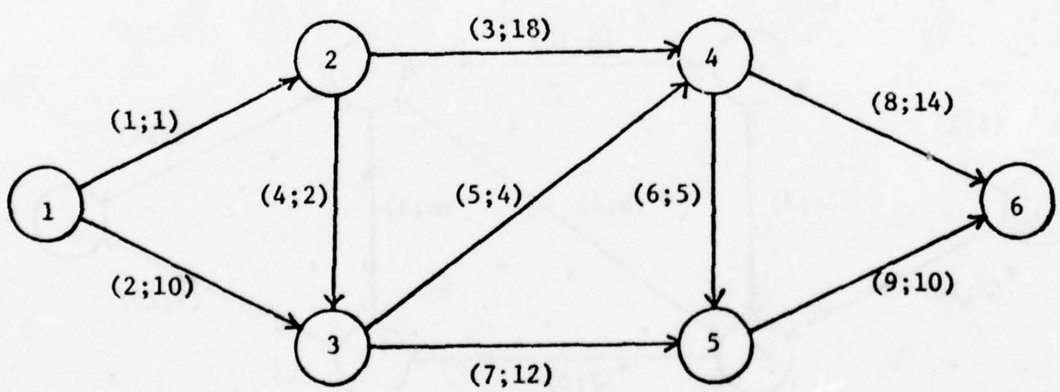


Figure E.7

Subnetwork for determining the eliminants of Activity 3 when $\theta = 2$.

The idea underlying the clusters is that they should be sets of activities whose effects on the subnetwork's duration are interrelated. Thus, if two clusters contain any activities in common, the activities in these two clusters all have an interrelated effect on the subnetwork's duration, so the two clusters are combined into one cluster. In the example clusters C_2 and C_3 both contain activity 5, so they are combined. The resulting clusters are

$$C_1 = \{2\} \text{ and } C_2 = \{5, 6, 7, 8, 9\} . \tag{6.10}$$

A non-critical activity may also influence the subnetwork's duration if its duration exceeds its mean.

Definition 3: An eliminant of a non-critical activity A is a critical activity which is not on the longest path when all activity durations are set to their means except for activity A which has its duration increased from m_A to $m_A + \theta s_A$ where θ is a nonnegative parameter.

For instance, if $\theta = 2$, the eliminants of the non-critical activity 3 in the example can be determined from Figure E.7. There the longest path is 1, 3, 6, and 9, so that the eliminants of activity 3 are the critical activities 2 and 7. In the example, when $\theta = 2$, none of the other non-critical activities (1, 4, 5, 6, and 8) have any eliminants. For a specified value of θ the eliminants of every non-critical activity are determined. If a non-critical activity A has eliminants, then the effect of A's eliminants on the subnetwork duration is related to A's duration, so A is added to every cluster containing at least one

of its eliminants. Thus in the example the clusters become

$$C_1 = \{2, 3\} \text{ and } C_2 = \{3, 5, 6, 7, 8, 9\} . \tag{6.11}$$

After the clusters have been expanded on the basis of eliminants, any two clusters containing common elements are combined. Therefore in the example, C_1 and C_2 are combined to form a single cluster

$$C_1 = \{2, 3, 5, 6, 7, 8, 9\} . \tag{6.12}$$

In general, after the determination of associates and eliminants for specified values of λ and θ and the subsequent combining of clusters, there may still be more than one cluster and some of the non-critical activities may not be in any cluster. Usually the larger the values of λ and θ the greater the number of activities in the clusters and the smaller the number of clusters. The clusters that remain represent sets of activities such that the effects on the subnetwork's duration of the activity durations for the activities within a set are all interrelated. Activities in different clusters have roughly independent effects on the subnetwork's duration. Activities not in any cluster have essentially no effect on the subnetwork's duration.

The consideration of critical activities, associates, eliminants, and the formation of clusters of related activities is obviously only one way of identifying the activities which have an important effect on the subnetwork's duration and their interrelationships. However, this particular procedure does have the following desirable properties:

<u>Property 1</u>: If $\lambda_2 > \lambda_1$, then any activity which would be an associate of a critical activity A when $\lambda = \lambda_1$ would also be an associate of A when $\lambda = \lambda_2$.

<u>Property 2</u>: If $\theta_2 > \theta_1$, then any critical activity which would be an eliminant of a non-critical activity A when $\theta = \theta_1$ would also be an eliminant of A when $\theta = \theta_2$.

<u>Property 3</u>: For any fixed value of λ, the set of activities in the union of the clusters is monotically nondecreasing as $\theta \to \infty$.

<u>Property 4</u>: The number of clusters is nonincreasing as $\theta \to \infty$.

<u>Property 5</u>: If $s_A > 0$ for a non-critical activity A, then there exists $\theta_A < \infty$ such that A will have some eliminants for any $\theta \geq \theta_A$.

<u>Property 6</u>: If $s_A > 0$ for every non-critical activity A and

$$\theta^* = \max\{\theta_A;\ A \text{ non-critical}\} ,$$

then for $\theta \geq \theta^*$ all activities will be in one cluster.

Most of these properties are fairly straightforward; however, Property 6 requires some special justification. This justification is based on the following definition and theorem which is proven in Sielken, Ringer, Hartley, and Arseven (1974).

<u>Definition 4</u>: In any acyclic network a <u>bridge</u> over any two consecutive arcs A_1 and A_2 is any arc A_3 such that all paths from the source to the sink passing through A_3 do not pass through either A_1 or A_2.

Theorem 1: In any acyclic network with no cut vertices there is at least one bridge for any pair of consecutive arcs.

Property 5 implies that all activities will belong to some cluster if $\theta \geq \theta^*$. Now consider any two consecutive activities A_1 and A_2 on the critical path. Theorem 1 implies that there is a bridge over A_1 and A_2, say A_3. Since the critical path passes through A_1 and A_2, A_3 cannot be on the critical path. Therefore, if $\theta \geq \theta^* \geq \theta_{A_3}$, A_1 and A_2 will be eliminants of A_3 and hence will be in the same cluster as A_3. Thus, since each cluster contains at least one original critical activity and any two consecutive critical path activities belong to the same cluster when $\theta \geq \theta^*$, there is only one cluster when $\theta \geq \theta^*$ and Property 6 is established.

6.3 Bounding the Discrete Subnetwork Duration Distribution F

6.3.1 Upper Bounds on F

Suppose that the cluster formation procedure described in subsection 6.2 has been carried out on a subnetwork for some specified values of θ and λ and yielded K clusters. For each cluster C so determined, let n_c be the number of activities in the cluster and let $v = 1, \ldots, 2^{n_c}$ index the 2^{n_c} configurations of activity durations corresponding to

(a) the duration for each activity A not in C being equal to its lower point ℓ_A, and

(b) the durations for the activities in C being at each of the 2^{n_c} possible combinations of their upper and lower points.

Then define

$$F^+(C;\ t) = \sum_{v=1}^{2^{n_c}} p_v I_t(t_v) \tag{6.13}$$

where p_v, t_v, and $I_t(t_v)$ are defined in (6.1), (6.4), and (6.5) respectively. The distribution function $F^+(C;\ t)$ is an upper bound on F. This can be shown by considering the following:

Theorem 2: For any cluster C, any t, and any activity A not in C,

$$F^+(C \cup \{A\};\ t) \leq F^+(C;\ t)\ .$$

(For the proof of this theorem, see Sielken, Hartley, and Spoeri (1975).) A straightforward application of Theorem 2 yields

Theorem 3: For any two clusters C_1 and C_2 and any t,

$$F^+(C_1 \cup C_2;\ t) \leq \min\{F^+(C_1;\ t),\ F^+(C_2;\ t)\}\ .$$

If C* represents the set (cluster) of all activities in the subnetwork, then

$$F(t) = F^+(C^*;\ t)\ . \tag{6.14}$$

Since C is a subset of C*, either Theorem 2 or Theorem 3 implies

$$F^+(C;\ t) \geq F(t) \tag{6.15}$$

for any cluster C.

Theorems 2 and 3 can also be used to define some tighter upper bounds on the subnetwork's duration distribution than $F^+(C;\ t)$. Historically, two different improved bounds have been employed, and both have been incorporated into the current subnetwork analysis

procedure. They are

$$F_1^+(t;\ \theta,\ \lambda) = F^+\left(\bigcup_{i=1}^{K} C_i;\ t\right) \tag{6.16}$$

and

$$F_2^+(t;\ \theta,\ \lambda) = \min_{1 \leq i \leq K} F^+(C_i;\ t)\ . \tag{6.17}$$

Let $F^+(t;\ \theta,\ \lambda)$ denote either $F_1^+(t;\ \theta,\ \lambda)$ or $F_2^+(t;\ \theta,\ \lambda)$. Then, since Property 2 of the cluster formation procedure implies that as θ increases the clusters expand or are combined, Theorems 2 and 3 imply that $F^+(t;\ \theta,\ \lambda)$ is a nonincreasing function of θ for every t and any λ. Property 6 and (6.14) imply that for $\theta \geq \theta^*$

$$F^+(t;\ \theta,\ \lambda) = F(t) \tag{6.18}$$

for every t and any λ. Also (6.14) along with the definitions (6.16) and (6.17) imply

$$F^+(t;\ \theta,\ \lambda) \geq F(t) \tag{6.19}$$

for all t, θ, and λ. These results are summarized in Theorem 4.

<u>Theorem 4</u>: (a) $F^+(t;\ \theta,\ \lambda)$ is a nonincreasing function of θ for every t and any λ;

(b) there exists a finite value θ^* such that $\theta \geq \theta^*$ implies $F^+(t;\ \theta,\ \lambda) = F(t)$ for every t and λ; and

(c) for any θ, λ, and t

$$F^+(t;\ \theta,\ \lambda) \geq F(t)\ .$$

6.3.2 Lower Bounds on F

Let n_c denote the number of activities in cluster C, and let $v = 1, \ldots, 2^{n_c}$ index the 2^{n_c} configuration of activity durations corresponding to

(a) the duration for each activity A not in the cluster being equal to its upper point u_A, and

(b) the durations for activities in the cluster being at each of the 2^{n_c} possible combinations of the upper and lower points.

Then define

$$F^-(C; t) = \sum_{v=1}^{2^{n_c}} p_v I_t(t_v) \tag{6.20}$$

where p_v, t_v, and $I_t(t_v)$ are as previously defined. Also define

$$F_1^-(t; \theta, \lambda) = F^-(\bigcup_{i=1}^{K} C_i; t) \tag{6.21}$$

and

$$F_2^-(t; \theta, \lambda) = \max_{1 \leq i \leq K} F^-(C_i; t) . \tag{6.22}$$

Using an argument completely analagous to that used to prove Theorem 4, Sielken, Hartley, and Spoeri (1975) also proved

<u>Theorem 5</u>: (a) $F^-(t; \theta, \lambda)$ is a nondecreasing function of θ for any fixed value of λ;

(b) there exists a finite value θ^* such that $\theta \geq \theta^*$ implies

$$F^-(t; \theta, \lambda) = F(t)$$

for every t and any λ; and

(c) for any θ, λ, and t

$$F^-(t;\ \theta,\ \lambda) \le F(t)\ .$$

(Again, $F^-(t;\ \theta,\ \lambda)$ is a generic term used to denote either $F_1^-(t;\theta,\lambda)$ or $F_2^-(t;\ \theta,\ \lambda)$.) Thus, $F^-(t;\ \theta,\ \lambda)$ is a valid lower bound on F.

6.3.3 The Tightness of the Bounds on F

That the F_1-bounds are tighter than the F_2-bounds can be seen as follows. The evaluation of $F_2^-(t;\ \theta,\ \lambda)$ involves the determination of $F^-(C_i;\ t)$ for each i whereas $F_1^-(t;\ \theta,\ \lambda) = F^-(\bigcup_{i=1}^{K} C_i,\ t)$. Let L_i be the length of the longest path when

1) the activities in C_i are at a particular configuration of their upper and lower durations and

2) all activities not in C_i have their upper durations.

Let L_U be the length of the longest path when

1) the configuration of upper and lower durations for the activities in C_i is the same as in the determination of L_i,

2) the activities in $\bigcup_{j=1}^{K} C_j - C_i$ are at <u>any</u> combination of their upper and lower durations, and

3) all activities not in $\bigcup_{j=1}^{K} C_j$ have their upper durations.

Then $L_i \ge L_U$ since every activity duration in the determination of L_i is greater than or equal to its corresponding duration in the determination of L_U. Since $L_i \ge L_U$ for any configuration of upper and lower durations for the activities in C_i,

$$F^-(\bigcup_{j=1}^{K} C_j;\ t) \ge F^-(C_i;\ t) \tag{6.23}$$

and

$$F_1^-(t;\ \theta,\ \lambda) = F^-(\bigcup_{j=1}^{K} C_j;\ t) \geq \max_{1\leq i\leq K} F^-(C_i;\ t) = F_2^-(t;\ \theta,\ \lambda)\ . \quad (6.24)$$

A similar argument can be used to show

$$F_1^+(t;\ \theta,\ \lambda) = F^+(\bigcup_{j=1}^{K} C_j;\ t) \leq \min_{1\leq i\leq K} F^+(C_i;\ t) = F_2^+(t;\ \theta,\ \lambda)\ . \quad (6.25)$$

The extent of the differences between the two upper bounds and two lower bounds depends heavily on the structure of the particular sub-network being analyzed and is a topic that should be considered in future empirical studies.

6.4 Using Sampling to Estimate the Upper and Lower Bounds on F

The only instance in which upper and lower bounds on F are computed rather than F itself is when it is computationally impractical to determine the longest path for each of the 2^n activity duration configurations.

For given θ and λ, the evaluation of $F_1^+(t;\ \theta,\ \lambda)$ only requires the determination of the longest path for each of 2^{n_U} activity configurations where n_U is the number of activities in the union of the clusters $C_o = \bigcup_{j=1}^{K} C_j$; i.e.,

$$n_U = \sum_{j=1}^{K} n_j\ . \quad (6.26)$$

The evaluation of $F_1^-(t;\ \theta,\ \lambda)$ also entails only 2^{n_U} longest path determinations. Likewise, the evaluation of $F_2^+(t;\ \theta,\ \lambda)$ or $F_2^-(t;\ \theta,\ \lambda)$ only requires the determination of the longest path for each of

$$n_s = \sum_{i=1}^{K} 2^{n_i} \quad (6.27)$$

activity configurations. Since 2^{n_U} is always greater than or equal to

n_s, $F_2^+(t;\ \theta,\ \lambda)$ and $F_2^-(t;\ \theta,\ \lambda)$ are the most economical bounds to compute in terms of the number of longest path determinations required. However, for any given θ and λ, $F_1^+(t;\ \theta,\ \lambda)$ and $F_1^-(t;\ \theta,\ \lambda)$ are tighter bounds than $F_2^+(t;\ \theta,\ \lambda)$ and $F_2^-(t;\ \theta,\ \lambda)$, respectively. Thus, in making the choice of which one of the two sets of bounds to compute, there is a trade-off between the accuracy of the bounds and the effort required to compute them.

Since the cluster formation procedure is such that the clusters expand or are pooled as θ increases, it may happen that for particular θ and λ, 2^{n_U} and 2^{n_i} for some i are both quite large even though θ is only moderately large. In this case it again becomes impractical to examine all the required activity configurations involved in determining either the F_1-bounds or the F_2-bounds. Consequently, if for the specified values of θ and λ, 2^{n_U} (or 2^{n_i} for some i, as the case may be) is excessively large, Subnetwork Analysis will compute estimates of the corresponding upper and lower bounds based on only a sample of the total number of possible configurations. The actual estimators of $F^+(C;t)$ or $F^-(C;t)$ based on sample values $x_1 \leq x_2 \leq \cdots \leq x_m$ for cluster C is

$$G(t) = \frac{\sum_{i=1}^{m} p_i I_t(x_i)}{\sum_{i=1}^{m} p_i} \tag{6.28}$$

where p_i and $I_t(\cdot)$ are as in (6.1) and (6.5) respectively. The sample values are determined by finding the longest path for each member of a systematic sample of the possible activity duration configurations for

the cluster. However, if the cluster contains more than 31 activities, the sample values are determined by random sampling. The estimator, $G(t)$, and the sampling procedures used in its evaluation are developed and discussed in Baker and Sielken (1978).

6.5 Estimating F by Extrapolating Between the Upper and Lower Bounds of F

Theorems 4 and 5 of subsection 6.3 imply that if $\theta_{i+1} \geq \theta_i$ and $\lambda_{i+1} \geq \lambda_i$ for all $i = 1, \ldots, I$ then

$$F^+(t;\, \theta_1,\, \lambda_1) \geq F^+(t;\, \theta_2,\, \lambda_2) \geq \ldots \geq F^+(t;\, \theta_I,\, \lambda_I) \geq F(t) \geq$$

$$F^-(t;\, \theta_I,\, \lambda_I) \geq F^-(t;\, \theta_{I-1},\, \lambda_{I-1}) \geq \ldots \geq F^-(t;\, \theta_1,\, \lambda_1) \tag{6.29}$$

for all t. Thus, if $F^+(t;\, \theta,\, \lambda)$ and $F^-(t;\, \theta,\, \lambda)$ have been calculated for I pairs $(\theta_i,\, \lambda_i)$ $i = 1, \ldots, I$ $(\theta_{i+1} \geq \theta_i,\, \lambda_{i+1} \geq \lambda_i)$, then $F(t)$ may be estimated by extrapolating between $F^+(t;\, \theta_I,\, \lambda_I)$ and $F^-(t;\, \theta_I,\, \lambda_I)$. As currently written, Subnetwork Analysis calculates upper and lower bounds on the subnetwork's approximate duration distribution for a sequence of three $(\theta,\, \lambda)$ pairs, $(\theta,\, \lambda) = (1, 1), (2, 2), (3, 2)$. An extrapolation procedure is then used to obtain an estimate of F. The procedure that has been developed for this purpose is documented in Baker and Sielken (1978).

6.6 A Summary of the Subnetwork Analysis Procedure

The following is a step-by-step description of the subnetwork analysis procedure in summary form. Recall that the objective of

Subnetwork Analysis is to determine an "approximation", say $\hat{F}$, to the subnetwork's duration distribution.

(a) If $n = 1$, let $\hat{F}$ be the actual activity duration distribution for the one activity comprising the subnetwork, and stop. Otherwise, go to Step b.

(b) Identify the two-point discrete distribution (ℓ_A, u_A, P_A, Q_A) for every activity A in the subnetwork.

(c) Ascertain the user's choice of

(1) NMAX, the maximum value of m for which all 2^m activity duration configurations are to be explicitly considered,

(2) the (θ, λ) pairs to be considered if not the standard pairs (1, 1), (2, 2), and (3, 2),

(3) whether the bounds on F are to be (F_1^-, F_1^+) or (F_2^-, F_2^+) if $n > \text{NMAX}$, and

(4) SAMSIZ, the sample size to be taken if, in the determination of bounds on F for some (θ, λ) pair, the number of activity configurations in the cluster being considered exceeds 2^{NMAX}.

(d) If the number of activities in the subnetwork doesn't exceed NMAX, compute the subnetwork's discrete duration distribution, F, explicitly, let $\hat{F} = F$, and stop. Otherwise, go to Step e.

(e) Do Steps f - i for every (θ, λ) pair. Then go to Step j.

(f) Form the clusters corresponding to (θ, λ). If the bounds are to be (F_1^-, F_1^+), go to Step g. If the bounds are to be (F_2^-, F_2^+), go to Step h.

(g) Form the union of the clusters and determine n_U. If $n_U \leq$ NMAX, evaluate the bounds (F_1^-, F_1^+) on the basis of all 2^{n_U} activity duration configurations. If $n_U >$ NMAX, take a sample of size SAMSIZ from the 2^{n_U} activity duration configurations and form both F_1^- and F_1^+ on the basis of this single sample. Go to Step e.

(h) Do the following for each cluster, C_i. Let n_i denote the number of activities in the cluster. If $n_i \leq$ NMAX, evaluate $F^-(C_i; t)$ and $F^+(C_i; t)$ on the basis of all 2^{n_i} activity duration configurations. If $n_i >$ NMAX, take a sample of size SAMSIZ from the 2^{n_i} activity duration configurations and form both $F^-(C_i; t)$ and $F^+(C_i; t)$ on the basis of this single sample.

(i) Form F_2^- and F_2^+ from the $F^-(C_i; t)$'s and $F^+(C_i; t)$'s respectively. Go to Step e.

(j) Form $\hat{F}$ by extrapolating the (F^-, F^+) bounds determined for the (θ, λ) pairs. Stop.

This process is repeated for every subnetwork in the simplified project network.

7. DECOMP: Decomposition

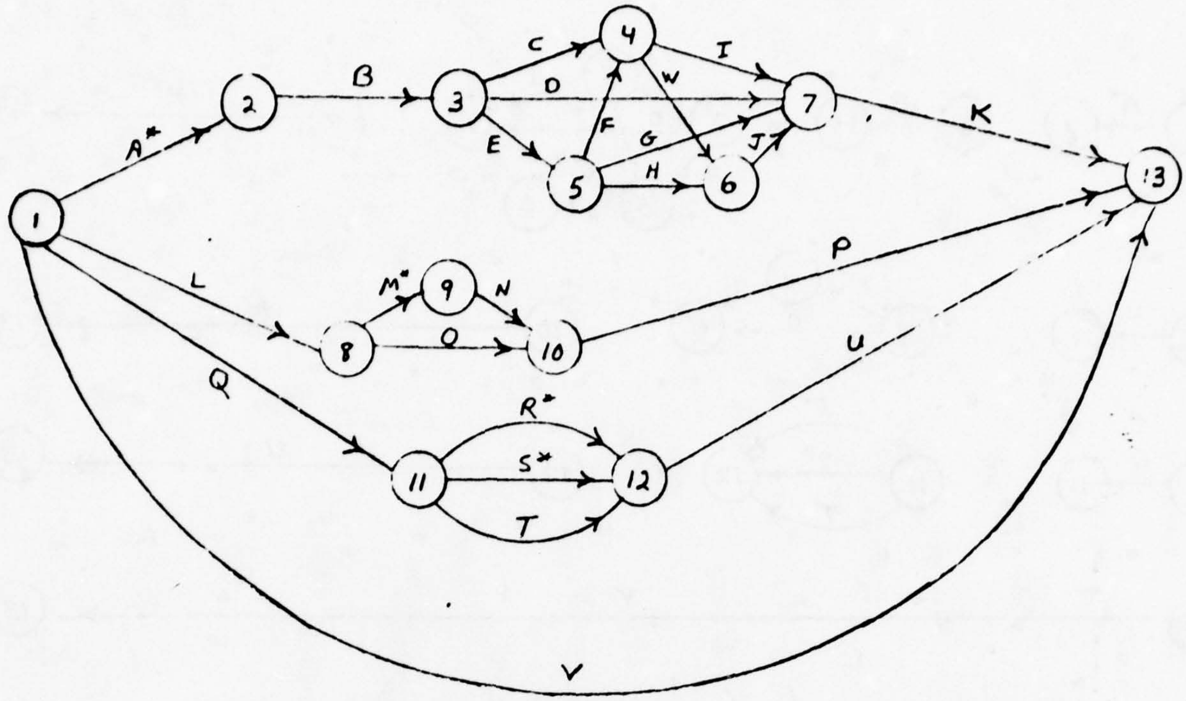
The objective of Deocmposition is to partition the simplified project network into the simplest possible subnetworks subject to the constaint that the subnetwork duration distributions can be easily combined to yield the project completion time distribution.

The simplified project network can be suitably partitioned using the following iterative procedure. First the simplified network is searched for subnetworks that begin at the beginning of the simplified network, end at the end of the simplified network, and are in parallel. Each parallel subnetwork is then subdivided into a sequence of smaller subnetworks that are in series. Each series subnetwork is then searched for parallel subnetworks. The partitioning into parallel and series subnetworks continues until no subnetwork can be further partitioned. This iterative partitioning procedure is illustrated in Figure E.8.

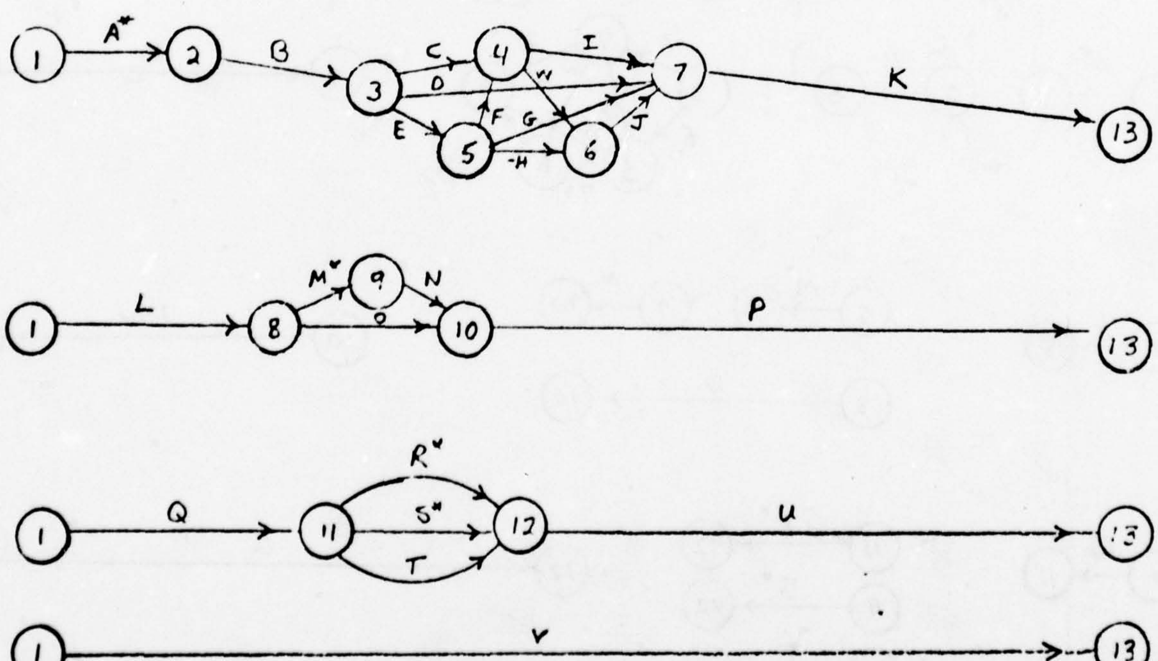
As in Simplification this partitioning of the simplified project network into subnetworks could have also included subnetwork configurations of the Wheatstone Bridge, Double Wheatstone Bridge, and Criss-Cross forms; however, the apparent frequency of these subnetwork configurations does not seem to justify the additional programming effort.

Since Steps 1 and 2 do not change the structure of the simplified project network and the partitioning of that network does not depend on the activity duration distributions, Decomposition is only done once and is really skipped when the general iterative algorithm returns to Step 3. Decomposition is documented in Sielken and Fisher (1976).

Figure E.8 The Decomposition of a Simplified Project Network*



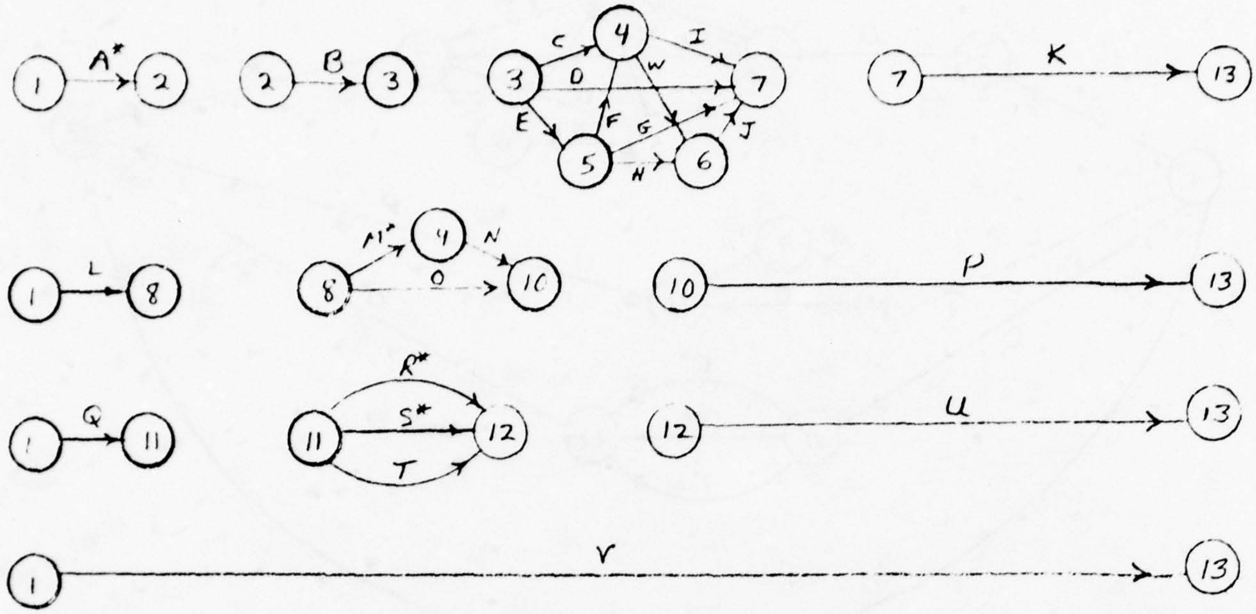
Simplified Project Network



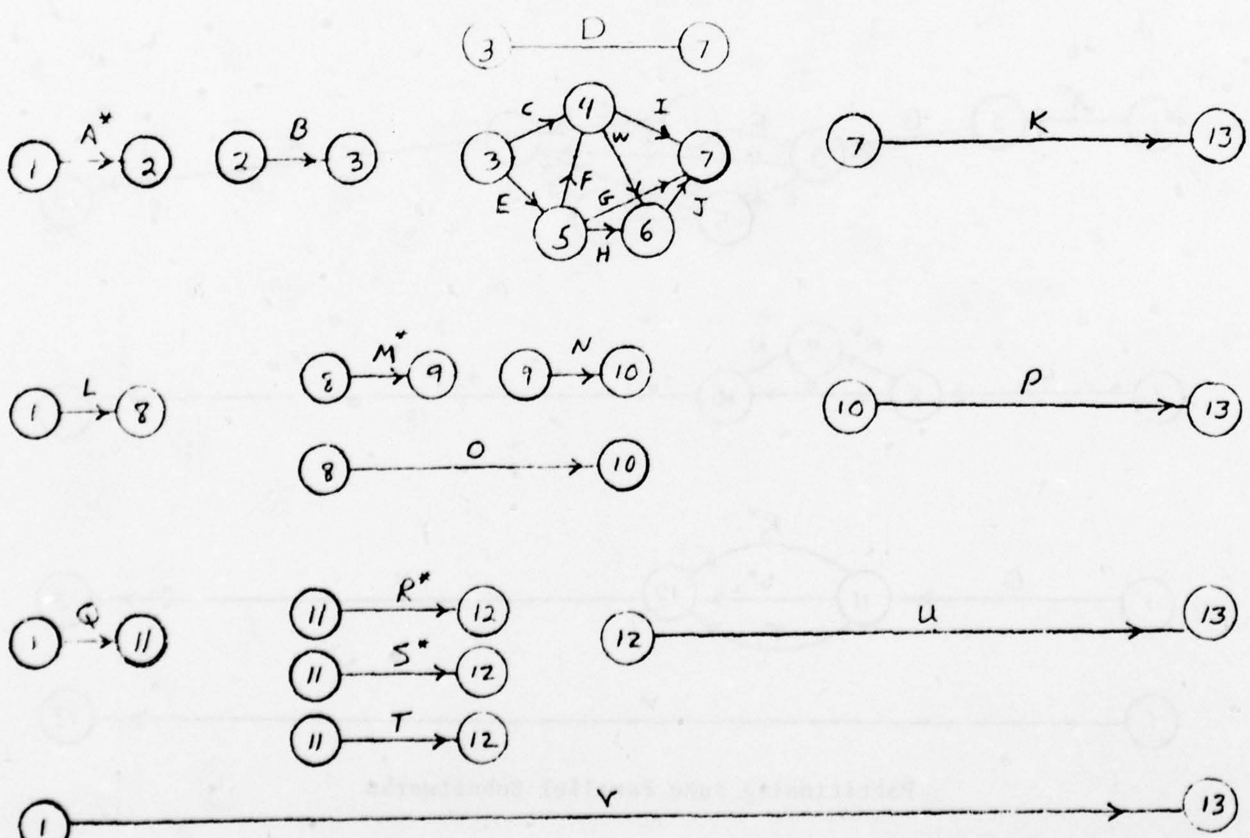
Partitioning into Parallel Subnetworks

The arcs A, M*, R*, and S* each represent non-simplifiable subnetworks and not single activities. Otherwise this "Simplified Project Network" could be further simplified.

Figure E.8 (Continued)



Partitioning into Series Subnetworks



Final Partitioned Simplified Project Network

8. SYNTH: Synthesis

The project's approximate completion time distribution can be determined by combining the approximate subnetwork duration distributions. When the project network is decomposed in Step 3, the result is a network of subnetworks with any two connected subnetworks being either in series or in parallel. Let SUB_1, and SUB_2 be any two such subnetworks, and let the corresponding approximate subnetwork duration distributions be $\hat{F}_1$ and $\hat{F}_2$. If SUB_1 and SUB_2 are in series, then the approximate duration distribution for SUB_1 and SUB_2 combined is

$$\hat{F}(t) = \sum_{s \leq t} \hat{F}_2(t - s)\hat{f}_1(s) = \sum_{s \leq t} \hat{F}_1(t - s)\hat{f}_2(s) \quad (8.1)$$

where $\hat{f}_1$ and $\hat{f}_2$ are the discrete probability density functions corresponding to $\hat{F}_1$ and $\hat{F}_2$ respectively. If SUB_1 and SUB_2 are in parallel, then the approximate duration distribution for SUB_1 and SUB_2 combined is

$$\hat{F}(t) = \hat{F}_1(t) \cdot \hat{F}_2(t). \quad (8.2)$$

By repeatedly combining subnetworks that are connected either in series or in parallel, the project's approximate completion time distribtuion is obtained.

Once the project's approximate completion time distribution has been determined, the project's approximate mean completion time, $\hat{T}$, can be calculated and compared with the project deadline. If the

project manager feels that $\hat{T}$ is sufficiently close to the project dealine, say within 5%, then the project schedule just determined in Step 1 is considered "optimal". Otherwise, a new project schedule must be determined by returning to Step 1 with a new TARGET TIME.

Step 1 only requires that the project would be completed by TARGET TIME if each activity's duration was exactly its mean duration. On the other hand, $\hat{T}$ takes into consideration the random nature of an activity's duration and hence will generally exceed TARGET TIME. The difficulty is in deciding how much less than the project deadline should TARGE TIME be in order that the corresponding $\hat{T}$ be sufficiently close to the project deadline. The algorithm iteratively updates its estimate of this TARGET TIME by

New TARGET TIME = Old TARGET TIME*(Project Deadline/$\hat{T}$).

The initial TARGET TIME would usually be the project deadline but could be chosen somewhat less than the project deadline. A typical sequence of TARGET TIMEs and $\hat{T}$'s for a project deadline of 400 might be

TARGET TIME = 400, $\hat{T}$ = 500,

TARGET TIME = 320, $\hat{T}$ = 360,

TARGET TIME = 356, $\hat{T}$ = 408.

Since the algorithm approximates the project's entire completion time distribution for each tentative schedule determined in Step 1,

the quantity $\hat{T}$ in the above discussion could just as easily be a specified percentile of the project completion time distribution. For example, the project manager might wish a minimum cost project schedule such that the probability of the project being completed before the project deadline is .90. In this case, the new TARGET TIMEs would be determined with $\hat{T}$ being the 90th percentile of the project's approximate completion time distribution instead of the project's approximate mean completion time.

Of course, each time Steps 1-5 are performed a new $\hat{T}$, approximate project completion time distribtion, and project cost are generated. This sequence of $\hat{T}$'s and project costs supplements the deterministic project cost curve in desribing the impact of the project deadline on the project cost.

Appendix F

Structure and Contents of the Temporary Data Sets

The project scheduling system requires eight temporary data sets which it uses to pass information from one program to the next. In the documentation, these data sets are called FILE0, FILE1, FILE2, ..., FILE7. The particular physical devices (tape, disk, etc.) which the file names represent are determined at execution time by the program's associated JCL-statements (Job Control Language statements) and by the user-defined unit numbers. The unit numbers corresponding to the eight data sets are defined symbolically in the FORTRAN code as F0, F1, F2, ..., F7, respectively. The particular values of the variables F0, ..., F7 are determined by DATA-statements which are inserted at the beginning of each program (see Appendix G for an example). Currently, F0 = 8, F1 = 9, F2 = 10, F3 = 11, F4 = 12, F5 = 13, F6 = 14, and F7 = 15. The particular form of the JCL-statements determine whether the unit numbers reference, tape, disk, or drum. The exact form of the JCL is highly dependent on the particular computer installation at which the system is being run. An example of the JCL used at Texas A&M is given in Appendix G.

The structure and contents of each of the data sets is outlined in the subsections that follow. In these subsections, variable names used in the input instructions (Appendix A) are not redefined. Any new variable names are defined when they are first used.

1. FILEO

Written by: MAIN(A,B), DPS(C); i.e., parts A and B of FILEO are written by MAIN and part C is written by DPS

Modified by: none

Used by: DPS(A), DSR(A,C), SIMP(A,B), MODSIMP(A,B): i.e., DPS uses only part A of FILEO, etc.

A. Record 1: $(4 \times MMAX + 5) \times 4$ bytes

1. NACT; (Integer, I * 4)
2. NODES: (I * 4)
3. NSRCE; (I * 4)
4. NSINK; (I * 4)
5. LNODEN; (I * 4)
6. S(MMAC,4); (two-dimensional array of I * 4 integers)

   For I = 1, ..., NACT,

   S(I,1) = the number of the I-th activity input, NA(I)

   S(I,2) = the I-th activity's origin node, NODEO(I)

   S(I,3) = the I-th activity's terminal node, NODET(I)

   S(I,4) = the number of times and costs for the I-th activity, NCT(I)

   For I > NACT,

   $S(I,1) = S(I,2) = S(I,3) = S(I,4) \equiv 0$

   Currently, MMAX = maximum number of activites in the project network = 1000.

B. Record 1: 4 bytes

1. JMAT; (I * 4)

C. Record 1: (NBREAK + 1) × 2 bytes

1. NBREAK; (Half-word integer, I * 2)

   NBREAK = the number of times and costs needed to specify the entire project's time-cost curve

2. BREAK(NBREAK); (a one-dimensional array of I * 2 integers)

   For I = 1, ..., NBREAK,

   BREAK(I) = the I-th largest time in the entire project's time-cost curve

   Currently, NBREAK $\leq$ 3000.

2. FILE1

Written by: MAIN(A)

Modified by: none

Used by: DPS(A), DSR(A)

A. Record 1: 8 bytes

1. TEST1; (I * 2)

2. TEST2; (I * 2)

3. TEST3; (I * 2)

TEST3 = a computation option in DPS that is defined internally by MAIN ≡ 0

4. STIME; (I * 2)

STIME = a parameter associated with TEST3 which is also defined internally by MAIN ≡ 0

* *

The remaining portion of FILE1 consists of one "block" of records for <u>each</u> original activity.

For I = 1, ..., NACT

Record (I + 1) . 1: (2 × NCT(I)) × 2 bytes

1. TIME(1); (I * 2)

2. TIME(2); (I * 2)

.
.
.

NCT(I). TIME(NCT(I)); (I * 2)

NCT(I) + 1. COST(1); (I * 2)

NCT(I) + 2. COST(2); (I * 2)

.
.
.

NCT(I) × 2. COST(NCT(I)); (I * 2)

For J = 1, ..., NCT(I) - 1

Record (I + 1). (J + 1); 20 bytes

1. IDIST; (I * 4)
2. PARM1; (Real, R * 4)
3. PARM2; (R * 4)
4. PARM3; (R * 4)
5. PARM4; (R * 4)

IDIST, PARM1, PARM2, PARM3, and PARM4 define the distribution (and its parameters) of the I-th activity on the J-th segment of the time-cost curve

* *

3. FILE2

Written by: DSR(A)

Modified by: DSR(A)

Used by: MODSIMP(A)

A. One record for each original activity

For I = 1, ..., NACT,

Record I: 20 bytes

1. JDIST; (I * 4)

JDIST = the distributional type of the I-th activity for the current TARGET TIME, TT

2. TPARM1; (R * 4)

3. TPARM2; (R * 4)

4. TPARM3; (R * 4)

5. TPARM4; (R * 4)

TPARM1, TPARM2, TPARM3, and TPARM4 are the I-th activity's distributional parameters for the current TARGET TIME, TT

4. FILE3

Written by: MAIN(A), SIMP(B)

Modified by: DECOMP(C)

Used by: DECOMP(B), SUBNET(A,C)

A. Record 1: 44 bytes

1. IFDF; (I * 4)
2. NMAX; (I * 4)
3. IPOOL; (I * 4)
4. SAMSIZ; (I * 4)
5. THELAM; (I * 4)
6. THETA(1); (R * 4)
7. THETA(2); (R * 4)
8. THETA(3); (R * 4)
9. LAMBDA(1); (R * 4)
10. LAMBDA(2); (R * 4)
11. LAMBDA(3); (R * 4)

B. Record 1: (4 × MMAX + 5) × 4 bytes

This record is an exact duplicate of FILE0 - part A except that the information refers to the simplified project network rather than the original project network. This record is destroyed by the execution of DECOMP and is replaced by part C.

C. Two records for each subnetwork identified by DECOMP

For K = 1, ..., number of subnetworks,

Record K.1: 4 bytes

1. M = the number of activities in the K-th subnetwork

Record K.2: $(3 \times M + 2) \times 4$ bytes

1. NSUB; (I * 4)

   NSUB = the number of the K-th subnetwork

2. NMM; (I * 4)

   NMM = the number of nodes in the K-th subnetwork

3. NET(3,M); (a two-dimensional array of I * 4 integers)

   For I = 1, ..., M,

   NET(1,I) = the origin node of the I-th activity in the K-th subnetwork

   NET(2,I) = the terminal node of the I-th activity in the K-th subnetwork

   NET(3,I) = the number assigned in the simplified project network to the K-th subnetwork's I-th activity

5. FILE4

Written by: MAIN(A), MODSIMP(B)

Modified by: MODSIMP(B), SYNTH(A), READFIL(A)

Used by: DSR(A), SUBNET(B), SYNTH(A)

A. Record 1: 12 bytes

1. NCYC; (I * 4)

NCYC = the number of the current iteration

2. TT; (R * 4)

TT = TARGET TIME for the current iteration

3. NFLAG; (I * 4)

NFLAG = internally defined parameter used by DSR ≡ 0 or 1

B. Record 1: 4 bytes

1. JNACT; (I * 4)

JNACT = the number of activities in the simplified project network

For I = 1, ..., JNACT,

Record I + 1: 96 bytes

1. CDFJ(12);(a one-dimenstional array of double word reals, R * 8)

CDFJ(1), ..., CDFJ(12) = $F^{-1}(.0)$, $F^{-1}(.05)$, $F^{-1}(.15)$, $F^{-1}(.25)$, ..., $F^{-1}(.95)$, $F^{-1}(1.00)$ where F is the cumulative distribution function for the I-th activity in the simplified project network.

6. FILE5

Written by: MAIN(A), DECOMP(B), SUBNET(C)

Modified by: SUBNET(C)

Used by: SYNTH(A,B,C)

A. Record 1: 16 bytes

1. IOPT; (I * 4)

2. NT; (I * 4)

3. PCT; (R * 4)

4. PD; (R * 4)

B. Record 1: 4 bytes

1. NOINS; (I * 4)

NOINS = the number of instructions generated by DECOMP for use in the synthesis process

Record 2; (27 × MAXINS) × 4 bytes

1. INSNO(MAXINS); (a one-dimension array of I * 4 integers)

2. ISUBNT(MAXINS); (a one-dimension array of I * 4 integers)

3. ISORP(MAXINS); (a one-dimension array of I * 4 integers)

4. JSUBNT(MAXINS, 24); (a two-dimensional array of I * 4 integers)

For I = 1, ..., NOINS,

INSNO(I) = the number of the I-th instruction

ISUBNT(I) = the number of the subnetwork resulting from the I-th instruction

ISORP(I) = indicates whether the subnetworks involved in the I-th instruction are in parallel or series

ISORP(I) = 0 implies the subnetworks are in series

ISORP(I) = 1 implies the subnetworks are in parallel

For J = 1, ..., 24,

JSUBNT(I,J) = the J-th subnetwork associated with the I-th instruction

C. One record for <u>each</u> subnetwork identified by DECOMP

For K = 1, ..., number of subnetworks,

Record K: (2 × JEDF + 2) × 4 bytes

1. ID; (I * 4)

   ID = the number of the K-th subnetwork

2. JEDF; (I * 4)

   JEDF = the number of subdivisions in the distribution function for the K-th subnetwork

3. XFD(JEDF,2); (a two-dimensional array of R * 4 reals)

   For J = 1, ..., JEDF,

   XFD(J,1) = the abscissa for the J-th tabled value of the cumulative distribution function for the K-th subnetwork

   XFD(J,2) = the ordinate for the J-th tabled value of the cumulative distribution function for the K-th subnetwork

7. FILE6

Written by: DPS(A)

Modified by: none

Used by: DSR(A)

A. One record for each segment of the entire project's time-cost curve

For K = 1, ..., NBREAK,

Record K: 4 × NACT bytes

1. XD(NACT,2); (a two-dimensional array of I * 2 integers)

For I = 1, ..., NACT

XD(I,1) = the direction (sign) of the change of the I-th activity's duration on the K-th function segment, i.e. +1, 0, -1

XD(I,2) = the maximum duration of the I-th activity on the K-th function segment

8. FILE7

Written by: SIMP(A)

Modified by: none

Used by: MODSIMP(A)

A. One record for each activity simplification operation identified by SIMP

For K = 1, ..., number of simplification operations,

Record K: 96 bytes

1. LIST(24); (a one-dimensional array of I * 4 integers)

For J = 1, ..., 24,

LIST(J) = a parameter which indentifies either the configuration or an activity involved in the K-th simplification operation

Table F.1

Contents of the Temporary Files at the Completion of Each Program

Program first iteration	FILE0	FILE1	FILE2	FILE3	FILE4	FILE5	FILE6	FILE7
MAIN	A,B	A	X	A	A	A	X	X
DPS	A,B,C	A	X	A	A	A	A	X
DSR	A,B,C	A	A	A	A	A	A	X
SIMP	A,B,C	A	A	A,B	A	A	A	A
MODSIMP	A,B,C	A	X	A,B	A,B	A	A	A
DECOMP	A,B,C	A	X	A,C	A,B	A,B	A	A
SUBNET	A,B,C	A	X	A,C	A	A,B,C	A	A
SYNTH	A,B,C	A	X	A,C	A	A,B	A	A
second & successive iterations								
DSR	A,B,C	A	A	A,C	A	A,B	A	A
MODSIMP	A,B,C	A	X	A,C	A,B	A,B	A	A
SUBNET	A,B,C	A	X	A,C	A	A,B,C	A	A
SYNTH	A,B,C	A	X	A,C	A	A,B	A	A

X means the file has no contents or that the information it does contain is never used in a subsequent step

Table F.2

Files Referenced by Each Program

Temporary Files*

PROGRAM	FILE0	FILE1	FILE2	FILE3	FILE4	FILE5	FILE6	FILE7
MAIN	X	X	X	X	X	X	X	X
DPS	X	X					X	
DSR	X	X	X		X		X	
SIMP	X			X				X
MODSIMP	X		X		X			X
DECOMP				X		X		
SUBNET				X	X	X		
SYNTH					X	X		
SAVEFIL	X	X	X	X	X	X	X	X
READFIL	X	X	X	X	X	X	X	X

* The temporary data sets referenced by each program in the computer system are indicated by an "X" in the appropriate column. Every data set referenced by a particular program should be defined for the job-step in which that program is executed.

Appendix G

Job Control Language for Texas A&M University Computer Facilities

1. Execution using FORTRAN source decks

A listing of the JCL used to generate the sample output of Appendix C is given below. During the first iteration each program in the sequence is compiled using the IBM FORTRAN G Compile and the load module which ultimately results is stored on the partitioned data set named USER.STAT.SIELKEN.PERT. Thus, on subsequent iterations the compile and link-edit steps are skipped.

```
//JOBLIB DD DSNAME=USER.STAT.SIELKEN.PERT.DISP=SHR
//MAIN EXEC FORTGCL.REGION=256K
//FORT.SYSIN DD *

        INSERT FORTRAN COURCE DECK FOR  MAIN  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(MAIN).DISP=SHR
//MAIN EXEC PGM=MAIN.REGION=320K.COND=(4.LT)
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA.DSN=&&FILE0.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT09F001     DD UNIT=SYSDA.DSN=&&FILE1.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT10F001     DD UNIT=SYSDA.DSN=&&FILE2.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT11F001     DD UNIT=SYSDA.DSN=&&FILE3.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT12F001     DD UNIT=SYSDA.DSN=&&FILE4.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT13F001     DD UNIT=SYSDA.DSN=&&FILE5.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT14F001     DD UNIT=SYSDA.DSN=&&FILE6.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT15F001     DD UNIT=SYSDA.DSN=&&FILE7.DISP=(NEW.PASS).
//         SPACE=(32000.(100.25)).DCB=(LRECL=3152.BLKSIZE=3156.RECFM=VBS)
//FT05F001 DD *

        INSERT DATA CARDS FOR  MAIN  HERE

//DPS EXEC FORTGCL.REGION=384K
//FORT.SYSIN DD *

        INSERT FORTRAN COURCE DECK FOR  DPS  HERE
```

```
//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(DPS),DISP=SHR
//DPS EXEC PGM=DPS,REGION=512K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001      DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT14F001      DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//DSR EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

          INSERT FORTRAN COURCE DECK FCR  DSR  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(DSR),DISP=SHR
//DSR EXEC PGM=DSR,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001      DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT10F001      DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001      DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT14F001      DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//SIMP EXEC FORTGCL,REGION=320K
//FORT.SYSIN DD *

          INSERT FORTRAN COURCE DECK FCR  SIMP  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(SIMP),DISP=SHR
//SIMP EXEC PGM=SIMP,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT11F001      DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT15F001      DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
//MODSIMP EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

          INSERT FORTRAN COURCE DECK FCR  MODSIMP  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(MODSIMP),DISP=SHR
//MODSIMP EXEC PGM=MODSIMP,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT10F001      DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001      DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT15F001      DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
//DECOMP EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

          INSERT FORTRAN COURCE DECK FCR  DECOMP  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(DECOMP),DISP=SHR
//DECOMP EXEC PGM=DECOMP,REGION=320K,CCND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT11F001      DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT13F001      DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SUBNET EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

          INSERT FORTRAN COURCE DECK FCR  SUBNET  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(SUBNET),DISP=SHR
//SUBNET EXEC PGM=SUBNET,REGION=448K,CCND=(4,LT)
//FT06F001 DD SYSOUT=A
```

```
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SYNTH EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

        INSERT FORTRAN COURCE DECK FCR  SYNTH  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(SYNTH),DISP=SHR
//SYNTH EXEC PGM=SYNTH,REGION=256K,CCNC=(4,LT)
//FT06F001 DD SYSOUT=A
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//DSR EXEC PGM=DSR,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001     DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT14F001     DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//MODSIMP EXEC PGM=MODSIMP,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT15F001     DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
//SUBNET EXEC PGM=SUBNET,REGION=448K,CCNC=(4,LT)
//FT06F001 DD SYSOUT=A
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SYNTH EXEC PGM=SYNTH,REGION=256K,CONC=(4,LT)
//FT06F001 DD SYSOUT=A
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
/*END
```

2. Execution using load modules

If the load modules corresponding to the programs in the system have already been prepared and stored on the disk file USER.STAT.SIELKEN.PERT., the following JCL is appropriate.

```
//JOBLIB DD DSNAME=USER.STAT.SIELKEN.PERT,DISP=SHR
//MAIN EXEC PGM=MAIN,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT09F001      DD UNIT=SYSDA,DSN=&&FILE1,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT10F001      DD UNIT=SYSDA,DSN=&&FILE2,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT11F001      DD UNIT=SYSDA,DSN=&&FILE3,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT12F001      DD UNIT=SYSDA,DSN=&&FILE4,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT13F001      DD UNIT=SYSDA,DSN=&&FILE5,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT14F001      DD UNIT=SYSDA,DSN=&&FILE6,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT15F001      DD UNIT=SYSDA,DSN=&&FILE7,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT05F001 DD *

          INSERT DATA CARDS FOR  MAIN  HERE

//DPS EXEC PGM=DPS,REGION=512K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001      DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT14F001      DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//DSR EXEC PGM=DSR,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001      DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT10F001      DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001      DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT14F001      DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//SIMP EXEC PGM=SIMP,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT11F001      DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT15F001      DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
//MODSIMP EXEC PGM=MODSIMP,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001      DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT10F001      DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001      DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT15F001      DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
```

```
//DECOMP EXEC PGM=DECOMP,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SUBNET EXEC PGM=SUBNET,REGION=448K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SYNTH EXEC PGM=SYNTH,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//DSR EXEC PGM=DSR,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT09F001     DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,PASS)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT14F001     DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,PASS)
//MODSIMP EXEC PGM=MODSIMP,REGION=320K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,PASS)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT15F001     DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,PASS)
//SUBNET EXEC PGM=SUBNET,REGION=448K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,PASS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
//SYNTH EXEC PGM=SYNTH,REGION=256K,COND=(4,LT)
//FT06F001 DD SYSOUT=A
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,PASS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,PASS)
/*END
```

Appendix H

Interrupting the Computer System

The number of iterations necessary to identify an "acceptable" project schedule using the new project scheduling procedure depends on many factors and usually cannot be determined in advance. Also the number of job steps allowed in the JCL-program for a given run varies widely from one computer installation to the next. (Currently, at Texas A&M the limitation is 10 job steps.) Thus, some provision must be made for interrupting the computer system prior to the identification of an acceptable project schedule and restarting it from the point of interruption.

One method is to let the "temporary" data sets defined in Appendix F correspond to user-supplied (permanent) storage devices rather than system-supplied (temporary) ones. This requires, however, that the user supply (and pay for)eight different permanent storage devices (or locations).

A second method which requires that the user supply only one permanent storage device involves the use of the programs SAVEFIL and READFIL which allow the user the capability of 1) interrupting the computer system at any of the starred points (*) indicated on Figure H.1 and 2) continuing the procedure from the point of interruption at some future time. SAVEFIL and READFIL are documented in subsections 1 and 2, respectively, of this appendix. Subsection 3 gives a listing of their associated JCL.

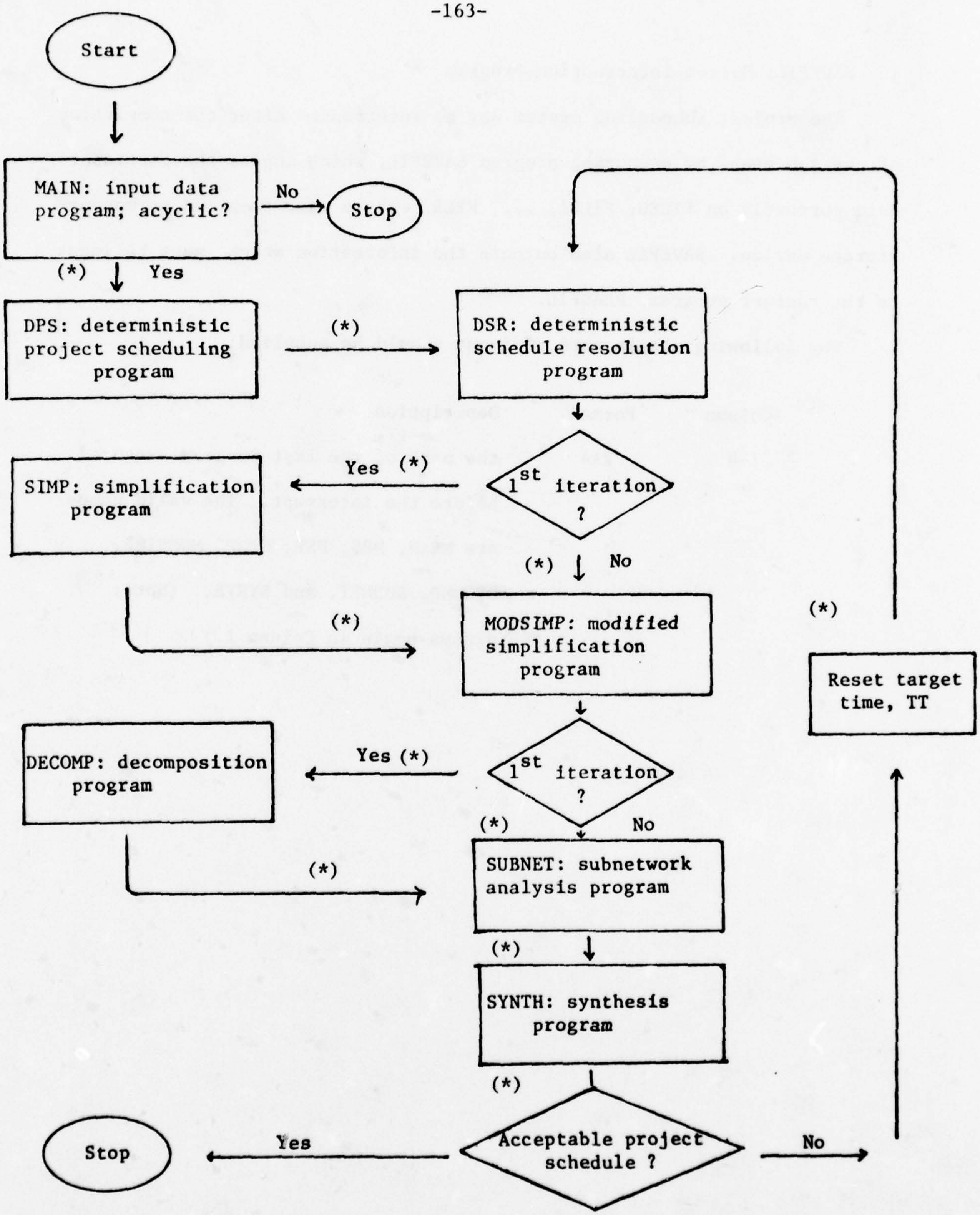


Figure H.1 Flowchart with (*) indicating where SAVEFIL may be used to interrupt the system.

1. SAVEFIL: System Interruption Program

The project scheduling system may be interrupted after the completion of any job step by executing program SAVEFIL, which copies the pertinent data currently on FILE0, FILE1, ..., FILE7 onto a user-supplied permanent storage device. SAVEFIL also outputs the information which must be input to the restart program, READFIL.

The following single card of input should be supplied:

Column	Format	Description
1-8	2A4	the name of the last program executed before the interrupt. The valid names are MAIN, DPS, DSR, SIMP, MODSIMP, DECOMP, SUBNET, and SYNTH. (Note: always begin in Column 1.)

2. READFIL: System Restart Program

When the project scheduling system has been interrupted by the use of SAVEFIL, READFIL must be used to restore the necessary information to the temporary data sets before the iterative procedure may resume. The following single card of input should be supplied:

Column	Format	Description
1-8	2A4	the name of the last program executed before the interrupt.
1-9	I2	0 if the procedure was interrupted before the first execution of SYNTH 1 otherwise

The remaining parameters should only be supplied if the procedure was interrupted immediately after the execution of SYNTH.

Column	Format	Description
11-20	F10.5	The value of TT for the next iteration if different from the value specified by SYNTH.
21-25	I5	1 if the values of any of the following are to be different from their current values in the next iteration: IOPT, PCT, PD 0 otherwise
26-30	I5	the new value for IOPT
31-40	F10.5	the new value for PCT
41-50	F10.5	the new value for PD

See Appendix A for the definition of TT, IOPT, PCT, and PD.

3. JCL for SAVEFIL and READFIL

The following exemplifies the JCL required by program SAVEFIL for the Texas A&M computer facilities. For this example, the user-supplied permanent storage device is named USER.STAT.SIELKEN.DATA.

```
//JOBLIB DD DSNAME=USER.STAT.SIELKEN.PERT,DISP=SHR
//SAVEFIL EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *
       INSERT FORTRAN SOURCE DECK FOR  SAVEFIL  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(SAVEFIL),DISP=SHR
//SAVEFIL EXEC PGM=SAVEFIL,REGION=256K,COND=(4,LT)
//FT04F001     DD DSN=USER.STAT.SIELKEN.DATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT08F001    DD UNIT=SYSDA,DSN=&&FILE0,DISP=(OLD,DELETE)
//FT09F001     DD UNIT=SYSDA,DSN=&&FILE1,DISP=(OLD,DELETE)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(OLD,DELETE)
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(OLD,DELETE)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(OLD,DELETE)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(OLD,DELETE)
//FT14F001     DD UNIT=SYSDA,DSN=&&FILE6,DISP=(OLD,DELETE)
//FT15F001     DD UNIT=SYSDA,DSN=&&FILE7,DISP=(OLD,DELETE)
//FT05F001 DD *
       INSERT DATA CARD FOR  SAVEFIL  HERE

/*END
```

Similarly, the JCL for program READFIL is as follows:

```
//JOBLIB DD DSNAME=USER.STAT.SIELKEN.PERT,DISP=SHR
//READFIL EXEC FORTGCL,REGION=256K
//FORT.SYSIN DD *

          INSERT FORTRAN SOURCE DECK FOR  READFIL  HERE

//LKED.SYSLMOD DD DSN=USER.STAT.SIELKEN.PERT(READFIL),DISP=SHR
//READFIL EXEC PGM=READFIL,REGION=256K,COND=(4,LT)
//FT04F001     DD DSN=USER.STAT.SIELKEN.DATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT08F001     DD UNIT=SYSDA,DSN=&&FILE0,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT09F001     DD UNIT=SYSDA,DSN=&&FILE1,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT10F001     DD UNIT=SYSDA,DSN=&&FILE2,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT11F001     DD UNIT=SYSDA,DSN=&&FILE3,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT12F001     DD UNIT=SYSDA,DSN=&&FILE4,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT13F001     DD UNIT=SYSDA,DSN=&&FILE5,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT14F001     DD UNIT=SYSDA,DSN=&&FILE6,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT15F001     DD UNIT=SYSDA,DSN=&&FILE7,DISP=(NEW,PASS),
//          SPACE=(32000,(100,25)),DCB=(LRECL=3152,BLKSIZE=3156,RECFM=VBS)
//FT05F001 DD *

          INSERT DATA CARD FOR  READFIL  HERE

//-------CONTINUE WITH THE JCL FOR THE NEXT PROGRAM IN THE SEQUENCE
```

Appendix I

Program Listings

1. MAIN

```
C      MAIN PROGRAM
C
C          THIS PROGRAM READS ALL USER SUPPLIED INPUT TO THE PROJECT
C          SCHEDULING SYSTEM AND STORES THE INFORMATION ON (SCRATCH)
C          DISK FILES DEFINED BY THE USER.  THE INFORMATION IS RETIEVED
C          AS NEEDED BY THE INDIVIDUAL PROGRAMS.
C
C          SEE TECHNICAL REPORT #57 FOR SPECIFIC INPUT INSTRUCTIONS
C
C
C          FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C          MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                     PROJECT NETWORK
C          LNMAX = THE LARGEST NODE NUMBER ALLOWED IN THE PROJECT NETWORK
C          NCTMAX = THE MAXIMUM NUMBER OF COMPLETION TIMES AND COSTS FOR
C                     EACH ACTIVITY
C          CURRENTLY, MMAX=1000; LNMAX=9999; NCTMAX=6
C
C      INTEGER*4 FILEO(4*MMAX+5),NA(MMAX)
C      INTEGER*4 NODEO(MMAX),NODET(MMAX),NCT(MMAX),IDIST(MMAX,NCTMAX-1)
C      INTEGER*2 TIME(MMAX,NCTMAX),COST(MMAX,NCTMAX)
C      REAL*4 PARM1(MMAX,NCTMAX-1),PARM2(MMAX,NCTMAX-1)
C      REAL*4 PARM3(MMAX,NCTMAX-1),PARM4(MMAX,NCTMAX-1)
C      EQUIVALENCE (FILEO(6),NA(1)),(FILEO(MMAX+6),NODEO(1))
C      EQUIVALENCE (FILEO(2*MMAX+6),NODET(1)),(FILEO(3*MMAX+6,NCT(1))
C          IN SUBROUTINE LOOP, THE ADDITIONAL ARRAY DIMENSIONS ARE:
C      DIMENSION A(LNMAX),B(LNMAX)
C
C          THE VALUES OF THE VARIABLES F0 - F7 SHOULD CORRESPOND TO UNIT
C          NUMBERS FOR THE INSTALLATION'S TEMPORARY STORAGE DEVISES(DISKS)
C          WHICH WILL BE REFERENCED BY THE PROJECT SCHEDULING SYSTEM.
C
       INTEGER F0/8/,F1/9/,       F3/11/,F4/12/,F5/13/
       INTEGER*4 SAMSIZ,THELAM,FILEO(4005),NA(1000)
       INTEGER*4 NODEO(1000),NODET(1000),NCT(1000),IDIST(1000,5)
       INTEGER*2 TIME(1000,6),COST(1000,6),TEST1,TEST2,TEST3,STIME
       REAL*4 PARM1(1000,5),PARM2(1000,5),PARM3(1000,5),PARM4(1000,5)
       REAL*4 THETA(3),LAMBDA(3)
       EQUIVALENCE (FILEO(1),NACT),(FILEO(2),NODES),(FILEO(3),NSRCE),
     *      (FILEO(4),NSINK),(FILEO(5),LNODEN),(FILEO(6),NA(1)),
     *      (FILEO(1006),NODEO(1)),(FILEO(2006),NODET(1)),
     *      (FILEO(3006),NCT(1))
       COMMON /BLKA/NACT,NODES,NSRCE,NSINK,LNODEN,NA,NODEO,NODET,NCT
       INTEGER NORM(5)/'NORM','AL',3*'  '/,BET(5)/'BETA',4*'  '/
       INTEGER RECT(5)/'RECT','ANGU','LAR(','UNIF','ORM)'/
       INTEGER FIXED(5)/'FIXE','D(CO','NSTA','NT)',' '/
       WRITE(6,2000)
2000   FORMAT(1H1,132('*')/1X,132('*')//////51X,'S T A T I S T I C A L
      *P E R T'/////)
       WRITE(6,2001)
2001   FORMAT(52X,'A PROJECT SCHEDULING PROCEDURE'///
      *60X,'WRITTEN AT THE '//55X,'INSTITUTE OF STATISTICS'/
      *57X,'TEXAS A&M UNIVERSITY'/53X,'COLLEGE STATION, TEXAS 77843'///
      *61X,'AUGUST 1978'/////33X,'INQUIRIES AND COMMENTS SHOULD BE ADDRES
      *SED TO: ROBERT L. SIELKEN JR.'//1X,132('*')/1X,132('*')//////)
C
C          GENERATE PARTS A AND B OF FILEO AND FILE1
C
       READ(5,110)NACT,NODES,NSRCE,NSINK,LNODEN
       DO 1300 I=1,NACT
```

```
      READ(5,110)NA(I),NODEO(I),NODET(I),NNCT
      NCT(I)=NNCT
      NNCT1=NNCT-1
      READ(5,110)(TIME(I,J),J=1,NNCT)
      READ(5,110)(COST(I,J),J=1,NNCT)
      DO 1300 L=1,NNCT1
      READ(5,150)J,IDIST(I,J),PARM1(I,J),PARM2(I,J),PARM3(I,J),PARM4(I,J
     *)
 1300 CONTINUE
      WRITE(F0)FILE0
      READ(5,110)TEST1,TEST2,JMAT
      WRITE(F0)JMAT
      ENDFILE F0
      TEST3=0
      STIME=0
      WRITE(F1)TEST1,TEST2,TEST3,STIME
      DO 1400 I=1,NACT
      NNCT=NCT(I)
      NNCT1=NNCT-1
      WRITE(F1)(TIME(I,J),J=1,NNCT),(COST(I,J),J=1,NNCT)
      DO 1400 J=1,NNCT1
      WRITE(F1)IDIST(I,J),PARM1(I,J),PARM2(I,J),PARM3(I,J),PARM4(I,J)
 1400 CONTINUE
      ENDFILE F1
C
C         GENERATE PART A OF FILE3
C
      READ(5,110)IEDF,NMAX,IPOOL,SAMSIZ
      READ(5,112)THELAM,(THETA(I),I=1,3),(LAMBDA(I),I=1,3)
      WRITE(F3)IEDF,NMAX,IPOOL,SAMSIZ,THELAM,(THETA(I),I=1,3),(LAMBDA(I)
     *,I=1,3)
C
C         GENERATE PART A OF FILE4 AND PART A OF FILE5
C
      READ(5,150)IOPT,NT,PCT,PD,TT
      NCYC=1
      NFLAG=0
      WRITE(F4)NCYC,TT,NFLAG
      WRITE(F5)IOPT,NT,PCT,PD
 110  FORMAT(8I10)
 112  FORMAT(I10,6F10.5)
 150  FORMAT(2I10,4F10.5)
C
C         OUTPUT THE PROBLEM AS IT WAS READ IN
C
      WRITE(6,1000)
 1000 FORMAT('0THIS IS THE OUTPUT FROM THE MAIN PROGRAM: MAIN'////
     *'0THE PROJECT NETWORK AS IT WAS READ IN:')
      WRITE(6,1001)NODES,NACT,NSRCE,NSINK,LNODEN
 1001 FORMAT(///'0THE NUMBER OF NODES IS ',I4,'.'/' THE NUMBER OF ACTIVI
     *TIES IS ',I4,'.'/' THE SOURCE NODE IS NUMBERED ',I5,' AND THE SINK
     * NODE IS NUMBERED ',I5,'.'/' THE LARGEST NODE NUMBER IS ',
     *I5,'.'//'1ACTIVITY ORIG TERM J TIME COST DISTRIBUTION',9X,'PARAMET
     *ERS'/)
      DO 1100 I=1,NACT
      NNCT=NCT(I)
      NNCT1=NNCT-1
      DO 1090 J=1,NNCT1
      ID=IDIST(I,J)+2
      GO TO (1020,1040,1060,1080),ID
 1020 IF(PARM4(I,J).EQ.-1.)GO TO 1030
```

```
      WRITE(6,1021)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *    BET,PARM1(I,J),PARM3(I,J),PARM2(I,J),PARM4(I,J)
 1021 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,
     *'MIN=',F13.5,', MAX=',F15.5,', ALPHA=',F12.5,', BETA=',F12.5)
      GO TO 1090
 1030 WRITE(6,1031)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *BET,PARM1(I,J),PARM2(I,J),PARM3(I,J)
 1031 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,
     *'MIN=',F13.5,', MODE=',F14.5,', MAX=',F14.5)
      GO TO 1090
 1040 IF(PARM4(I,J).EQ.-1.)GO TO 1050
      WRITE(6,1041)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *    NORM,PARM2(I,J),PARM3(I,J)
 1041 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,
     *'MEAN=',F12.5,', ST.DEV=',F12.5)
      GO TO 1090
 1050 WRITE(6,1051)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *    NORM,PARM1(I,J),PARM2(I,J),PARM3(I,J)
 1051 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,
     *'MIN=',F13.5,', MEAN=',F14.5,', MAX=',F14.5)
      GO TO 1090
 1060 WRITE(6,1061)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *    RECT,PARM1(I,J),PARM3(I,J)
 1061 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,
     *'MIN=',F13.5,', MAX=',F15.5)
      GO TO 1090
 1080 WRITE(6,1081)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J),
     *    FIXED,PARM1(I,J)
 1081 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5,1X,5A4,1X,'TIME=',F12.5)
 1090 CONTINUE
      J=NNCT
      WRITE(6,1095)I,NODEO(I),NODET(I),J,TIME(I,J),COST(I,J)
 1095 FORMAT(3X,I4,2X,I5,    I5,1X,I1,I5,I5)
 1100 CONTINUE
      IF(TEST1)1104,1104,1106
 1104 WRITE(6,1105)
 1105 FORMAT(////'0THE INFORMATION INPUT TO THE DETERMINISTIC PROJECT SC
     *HEDULING PROGRAM WILL BE PRINTED.')
      GO TO 1110
 1106 WRITE(6,1107)
 1107 FORMAT(////'0THE INFORMATION INPUT TO THE DETERMINISTIC PROJECT SC
     *HEDULING PROGRAM WILL NOT BE PRINTED.')
 1110 IF(TEST2)1114,1114,1116
 1114 WRITE(6,1115)
 1115 FORMAT(' THE INTERMEDIATE OUTPUT GENERATED BY THE DETERMINISTIC PR
     *OJECT SCHEDULING PROGRAM WILL BE PRINTED.')
      GO TO 1120
 1116 WRITE(6,1117)
 1117 FORMAT(' THE INTERMEDIATE OUTPUT GENERATED BY THE DETERMINISTIC PR
     *OJECT SCHEDULING PROGRAM WILL NOT BE PRINTED.')

 1120 IF(JMAT)1128,1128,1121
 1121 WRITE(6,1122)
 1122 FORMAT('0THE SIMPLIFICATION PROGRAM WILL PRINT ONLY THE MINIMUM CU
     *TPUT.')
      GO TO 1130
 1128 WRITE(6,1129)
 1129 FORMAT('0THE SIMPLIFICATION PROGRAM WILL PRINT THE COMPLETE OUTPUT
     *.')
 1130 CONTINUE
      WRITE(6,1131)IEDF,NMAX,SAMSIZ
```

```
1131  FORMAT('ODURING THE SUBNETWORK ANALYSIS PROGRAM,'/
     *5X,'THE NUMBER OF SUBDIVISIONS IN THE ESTIMATED DURATION DISTRIBUT
     *ION FOR EACH SUBNETWORK WILL BE ',I4,';'/
     *5X, 'THE LARGEST NUMBER OF ACTIVITIES IN A SUBNETWORK FOR WHICH TH
     *E SUBNETWORK',1H','S'/10X, 'EXACT DISCRETE DURATICN DISTRIBUTION W
     *ILL BE ENUMERATED IS ',I4,';'/
     *5X,  'THE MAXIMUM NUMBER OF ACTIVITY DURATION CONFIGURATIONS EXPLI
     *CITLY CONSIDERED IN THE DETERMINATION OF THE'/10X,'UPPER AND LOWER
     * BOUNDS ON EACH SUBNETWORK',1H','S DISCRETE DURATION DISTRIBUTION
     *WILL BE ',I5,';')
      IF(IPCOL)1132,1132,1134
1132  WRITE(6,1133)
1133  FORMAT(5X,'THE MAXIMUM-CLUSTER PROCEDURE WILL BE USED IN THE DETER
     *MINATION OF THE BOUNDS;')
      GO TO 1136
1134  WRITE(6,1135)
1135  FORMAT(5X,'THE UNION-CLUSTER PROCEDURE WILL BE USED IN THE DETERMI
     *NATION OF THE BOUNDS;')
1136  WRITE(6,1137)
1137  FORMAT(5X,'THE FOLLOWING (THETA,LAMBDA) PAIRS WILL BE USED IN THE
     *FORMATION OF THE CLUSTERS:')
      IF(THELAM)1138,1138,1140
1138  WRITE(6,1139)
1139  FORMAT(10X,'(    1.00000,    1.00000),(    2.00000,    2.00000),(    3.
     *00000,    2.00000).')
      GO TO 1150
1140  WRITE(6,1141)(THETA(I),LAMBDA(I),I=1,3)
1141  FORMAT(10X,'(',F10.5,',',F10.5,'),(',F10.5,',',F10.5,'),(',F10.5,
     *',',F10.5,').')
1150  WRITE(6,1151)NT
1151  FORMAT('OTHE NUMBER OF SUBDIVISICNS IN THE NETWORK',1H','S SYNTHES
     *IZED COMPLETION TIME DISTRIBUTION WILL BE 'I2,'.')
      IF(IOPT)1160,1160,1165
1160  WRITE(6,1161)PCT,PD
1161  FORMAT('OTHE ',F7.2,'-TH PERCENTILE OF THE NETWORK',1H','S APPROXI
     *MATE COMPLETION TIME DISTRIBUTION WILL BE COMPARED TO THE'/
     *10X,'SPECIFIED PROJECT DEADLINE OF ',F10.5,'.')
      GO TO 1170
1165  WRITE(6,1166)PD
1166  FORMAT('OTHE MEAN OF THE NETWORK',1H','S APPROXIMATE COMPLETION TI
     *ME DISTRIBUTION WILL BE COMPARED TO THE'/
     *10X,'SPECIFIED PROJECT DEADLINE OF ',F10.5,'.')
1170  WRITE(6,1171)TT
1171  FORMAT('OTHE TARGET TIME FOR THE FIRST ITERATION IS ',F10.5,'.'//)
C
C        CHECK FOR LOOPS IN THE PROJECT NETWORK
C
      CALL LOOP
      STOP
      END
      SUBROUTINE LOOP
C
C        THIS PROGRAM DETERMINES WHETHER THE PROJECT NETWORK IS ACYCLIC
C        OR CONTAINS NODES WHICH ARE PART OF LOOPS (CYCLES)
C
      IMPLICIT INTEGER*4 (A-Z)
      DIMENSION A(9999),B(9999)
      [illegible] NA(1000),TAIL(1000),HEAD(1000),NCT(1000)
      [illegible] NSINK,LNODEN,NA,TAIL,HEAD,NCT

      [illegible]
```

```
      WRITE(6,2004)
 2004 FORMAT('OTHIS IS THE OUTPUT FROM SUBPROGRAM LOOP. THE PROGRAM SEAR
     *CHES THE GIVEN PROJECT NETWORK FOR LOOPS (CYCLES).'/' A VALID ',
     *'PROJECT NETWORK SHOULD CONTAIN NO LOOPS.'///)
      DO 4 J=1,LNODEN
      A(J)=0
 4    B(J)=0
C     FORM THE 1ST HIERARCHY
      INODE = 0
   80 HIER = 2
 82   INODE=INODE+1
      IF(INODE.EQ.NSRCE.OR.INODE.EQ.NSINK)GO TO 81
      DO 5 I=1,M
      IF(INODE.EQ.TAIL(I))GO TO 81
 5    CONTINUE
      GO TO 82
 81   J=0
      DO 1 I= 1,M
      IF(HEAD(I).NE.INODE) GO TO 1
      J = J+1
      A(J)=TAIL(I)
      IF (TAIL(I).EQ.INODE) GO TO 998
    1 CONTINUE
      IF (J.EQ.0) GO TO 997
      IA=J
      J=0
C     FORM THE SUBSEQUENT HIERARCHIES
  102 CONTINUE
      DO 2 II=1,IA
      DO 3 I=1,M
      IF (HEAD(I).NE.A(II)) GO TO 3
      IF(TAIL(I).EQ.INODE) GO TO 998
      IF(J.EQ.0) GO TO 40
      DO 10 K=1,J
      IF(TAIL(I).EQ.B(K)) GO TO 11
 10   CONTINUE
 40   CONTINUE
      J=J+1
      B(J)=TAIL(I)
 11   CONTINUE
    3 CONTINUE
    2 CONTINUE
      IF (J.EQ.0) GO TO 997
      HIER = HIER+1
      IA=J
      J = 0
      DO 20 I=1,IA
 20   A(I)=B(I)
      GO TO 102
  997 CONTINUE
      WRITE(6,2002) INODE
 2002 FORMAT(' NODE',I5,' IS NOT PART OF A LOOP')
      IF (INODE.NE.LNODEN)GO TO 80
      IF(ILOOP.EQ.1) GO TO 50
      WRITE(6,1000)
 1000 FORMAT (' THERE ARE NO LOOPS IN THIS NETWORK')
      GO TO 999
 50   WRITE(6,51)
 51   FORMAT(' THERE ARE NO OTHER LOOPS IN THIS NETWORK')
      A(1000000000) = 0.
      GO TO 999
```

```
  998 WRITE (6,1001) INODE,HIER
 1001  FORMAT(' NODE ',I5,'IS PART OF A LOOP.  THERE ARE ',I3,' ACTIVITIE
      *S IN THE LOOP.')
       ILOOP=1
       INODO=INODE+1
       IF(INODO.NE.LNODEN)GO TO 80
  999  RETURN
       END
```

2. DPS

```
C       DETERMINISTIC PROJECT SCHEDULING PROGRAM
C
C     DETERMINISTIC PROJECT SCHEDULER:  THIS PROGRAM COMPUTES THE
C     MINIMUM PROJECT COST AND ASSOCIATED OPTIMAL FIXED ACTIVITY TIMES
C     FOR EVERY FEASIBLE PROJECT DEADLINE.
C-
C        ALL VARIABLES ARE INTEGER.
C       (IF ANY VARIABLE IS NOT ALREADY IN INTEGER FORM,  THE VALUES MUST
C       BE RESCALED - THAT IS, MULTIPLIED BY AN APPROPRIATE POWER OF 10 -
C       UNTIL THE VALUES ARE INTEGER.)
C
C*********************************************************************
C
C       DEFINITION OF VARIABLES:
C
C          ABAR(I,J) = TIME(I,NK(I)+1-J) + XNODE(ORIG(I))-XNODE(TERM(I))
C             C(I,J) = DECREASE IN I TH ACT'S COST PER UNIT FOR J TH TIME
C                CAP = MIN(FLOW REACHING ORIGIN NODE, EXCESS CAPACITY TO
C                         TERMINAL NODE)
C          COST(I,J) = COST OF COMPLETING ACTIVITY I AT TIME(I,J)
C                DEL = MIN(DELTA1,DELTA2)
C             DELTA1 = MIN(-ABAR(I,J) WITH I LABELED AND J UNLABELED,
C                         ABAR(I,J)<0)
C             DELTA2 = MIN(ABAR(I,J) WITH I UNLABELED AND J LABELED,
C                         ABAR(I,J)>0)
C            DIREC(J) = DIRECTION OF FLOW REACHING NODE J
C                         (0=FORWARD, 1=REVERSE)
C          FLOW(I,J) = FLOW IN J TH PIECE OF ACTIVITY I
C                INF = ANY NUMBER GREATER THAN MAX(CAP)
C                         (CURRENTLY SET AT (2*MAX +1))
C              K1(I) = THE NUMBER OF THE TIME-COST PIECE USED IN
C                         LABELING TERM(I) FROM ORIG(I)
C              KOUNT = KEEPS TRACK OF ORDER IN WHICH NODES WERE LABELED
C           LABEL(I) = 0 IF NODE I UNLABELED
C                      1 IF NODE I LABELED
C             LINPUT = SPECIFIC VALUE OF LAMBDA IF TEST3=1 OR 2
C                 NA = TOTAL NUMBER OF ACTIVITIES
C              NK(I) = NUMBER OF DIFFERENT TIMES AND COSTS FOR ACTIVITY I
C                 NN = TOTAL NUMBER OF NODES
C            ORIG(I) = ORIGIN NODE FOR ACTIVITY I
C           ORIG2(I) = WHERE THE FLOW IS FROM - USED IN LABELING ONLY
C              PCOST = PROJECT COST FUNCTION
C               SINK = NUMBER OF THE SINK NODE
C             SOURCE = NUMBER OF THE SOURCE NODE
C            TERM(I) = TERMINAL NODE FOR ACTIVITY I
C              TEST1 = OPTION TO SUPPRESS PRINTING OF INPUT
C                         (0=PRINT, 1=NO PRINT)
C              TEST2 = OPTION TO SUPPRESS INTERMEDIATE OUTPUT
C                         (0=PRINT, 1=NO PRINT)
C              TEST3 = OPTION TO SPECIFY VALUE FOR LAMBDA
C                         (0=NO, 1=YES AND SEE INTERMEDIATE OUTPUT,
C                         2=YES BUT NO INTERMEDIATE OUTPUT)
C          TIME(I,J) = J TH BREAKPOINT (DURATION TIME) FOR ACTIVITY I
C            XACT(I) = ACTIVITY DURATION TIME
C           XNODE(I) = NODE TIME
C           XDIFF(I) = XNODE(ORIG(I))-XNODE(TERM(I)),  AN UPPER BOUND ON
C                         THE ACTIVITY DURATION TIME
C      I,J,K,M,N,P = INDICES
C     INODE,ITERM,IACT,IORIG,IDIFF,ETC.
C                  = NON-INDEXED VERSIONS OF XNODE(I),TERM(I),XACT(I),
```

```
C                        ORIG(I),XDIFF(I),ETC.
C
C***********************************************************************
C
C     FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C     MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                 PROJECT NETWORK
C     LNMAX = THE LARGEST NODE NUMBER ALLOWED IN THE PROJECT NETWORK
C     BKMAX = THE MAXIMUM NUMBER OF BREAK POINTS ALLOWED IN THE
C                 ENTIRE PROJECT'S TIME-COST CURVE
C     NCTMAX = THE MAXIMUM NUMBER OF COMPLETION TIMES AND COSTS FOR
C                 EACH ACTIVITY
C     CURRENTLY, MMAX=1000; LNMAX=9999; BKMAX=3000; NCTMAX=6
C
C     DIMENSION CAP(LNMAX),FLOW(MMAX,NCTMAX),C(MMAX,NCTMAX),ORIG(MMAX),
C    *          TERM(MMAX),TIME(MMAX,NCTMAX),COST(MMAX,NCTMAX),NK(MMAX),
C    *          ABAR(MMAX,NCTMAX),XDIFF(MMAX),XNODE(LNMAX),XACT(MMAX),
C    *          DIREC(LNMAX),LABEL(LNMAX),K1(LNMAX),ORIG2(LNMAX),
C    *          KOUNT(LNMAX),AORD(MMAX),ND(LNMAX),NDD(LNMAX),IP(MMAX),
C    *          CTIME(MMAX),FILE0(BKMAX),FILE6(2,MMAX),NUMS(MMAX)
C
C***********************************************************************
C
      IMPLICIT INTEGER*2(A-Z)
      INTEGER*4 NA,NN,ORIG,TERM,NK,I,SOURCE,SINK,NUMS(1000),LNODEN
      REAL*4 CAP(9999),FLOW(1000, 6),C(1000, 6),PCOST,INF,PCOST1,TT,
     1KCOST,ACOST ,PNEW
      COMMON ORIG,TERM,NK,NA,LNODEN,TIME,CTIME,XNODE,AORD,LMIN,LMAX,
     *       TEST1
      DIMENSION ORIG(1000),TERM(1000),TIME(1000, 6),COST(1000, 6),
     *NK(1000),ABAR(1000, 6),XDIFF(1000),XNODE(9999),XACT(1000),
     *K1(9999),ORIG2(9999),KOUNT(9999),AORD(1000),CTIME(1000),
     *DIREC(9999),LABEL(9999),ND(9999),NDD(9999),IP(1000)
      INTEGER F0/8/,F1/9/,                          F6/14/
      DIMENSION FILE0(2000),FILE6(2,1000)
      WRITE(6,1500)
 1500 FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE DETE
     *RMINISTIC PROJECT SCHEDULER: DPS'//1X,132('*')/1X,132('*'))
C
C     INPUT DATA
C
      REWIND F0
      REWIND F1
      KKKK=0
      READ(F0)NA,NN,SOURCE,SINK,LNODEN,NUMS,ORIG,TERM,NK
      INF=0.
      PCOST=0.
      READ(F1)TEST1,TEST2,TEST3,LINPUT
      DO 12 I=1,NA
      KN=NK(I)
      READ(F1)(TIME(I,J),J=1,KN),(COST(I,J),J=1,KN)
      KN1=KN-1
      DO 12 J=1,KN1
      READ(F1)
   12 CONTINUE
      IF(TEST1.EQ.1) GO TO 401
      WRITE(6,150) NN,NA,SOURCE,SINK
  401 CALL ORDER
C
C     SET UP INITIAL VALUES
C
```

```
      IF(TEST1.EQ.1) GC TO 193
      K3=1
  192 K2=K3+8
      IF(K2.GT.LNUDEN)K2=LNCDEN
      WRITE(6,151) (K,K=K3,K2)
      WRITE(6,157) (XNODE(K),K=K3,K2)
      IF(K2.GE.LNUDEN)GO TO 191
      K3=K2+1
      GO TO 192
  191 WRITE(6,152)
  193 DC 10 I=1,NA
      LABEL(I)=0
      XDIFF(I)=XNCDE(ORIG(I))-XNCDE(TERM(I))
      NKM1=NK(I)-1
      KN=NK(I)
      DO 9 J=1,NKM1
      TT=TIME(I,J+1)-TIME(I,J)
      IF(TT)7,8,7
    7 C(I,J)=(COST(I,J)-COST(I,J+1))/TT
      GC TC 6
    8 C(I,J)=0.
    6 IF(INF.LT.C(I,J))  INF=C(I,J)
      XACT(I)=XDIFF(I)
      IF(XACT(I).LT.TIME(I,J+1)) XACT(I)=TIME(I,J+1)
      JJ=NK(I)-J+1
      ABAR(I,J)=TIME(I,JJ)+XDIFF(I)
      FLOW(I,J)=0
    9 CONTINUE
      FILE6(1,I)=0
      FILE6(2,I)=XACT(I)
      ABAR(I,KN)=TIME(I,1)+XDIFF(I)
      FLOW(I,KN)=0
      IF(TEST1.EQ.1) GO TO 10
      WRITE(6,153)NUMS(I),XACT(I),ORIG(I),TERM(I),(J,TIME(I,J),COST(I,J)
     *,C(I,J),ABAR(I,J),J=1,NKM1)
      WRITE(6,156) KN,TIME(I,KN),COST(I,KN),ABAR(I,KN)
   10 CCNTINUE
      WRITE(F6)((FILE6(I,J),I=1,2),J=1,NA)
      INF=2.*INF+1.
      DO 417 I=1,NA
      C(I,NK(I))=0.
      NKM1=NK(I)-1
      PCCST1=0.
      IKK=0
      DO 418 K=1,NKM1
      IF(K.NE.1) GO TO 40
      XIJ=XACT(I)
      IF(XIJ.GT.TIME(I,2)) XIJ=TIME(I,2)
      GO TO 41
   40 XIJ=XACT(I)-TIME(I,K)
      IF(XIJ.LT.0) XIJ=0
      IF(XIJ.GT.(TIME(I,K+1)-TIME(I,K))) XIJ=TIME(I,K+1)-TIME(I,K)
      IF(IKK.EC.1) GO TO 41
      IF(C(I,K).GT.C(I,K-1)) GO TO 50
   41 PCOST1=PCCST1+C(I,K)*XIJ
      GC TO 418
   50 IKK=1
      WRITE(6,237) I,I
      PCOST1=PCOST1+C(I,K)*XIJ
  418 CUNTINUE
      PCCST=PCOST+COST(I,1)+C(I,1)*TIME(I,1)-PCOST1
```

```
      PNEW=PCOST
  417 CONTINUE
      LAMBDA=LMAX
      KKKK=KKKK+1
      FILEO(KKKK)=LAMBDA
      IF (TEST3.GE.1) GO TO 700
      WRITE(6,154)
      LINPUT=0
      GO TO 96
 700  CONTINUE
      IF(LINPUT.LT.LMIN) GO TO 705
      IF(LINPUT.GE.LMAX) GO TO 704
      IF (TEST3.EQ.2) GO TO 724
      WRITE(6,155) LINPUT
   96 WRITE(6,200) LAMBDA, PCOST
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 724
      WRITE(6,235)
  724 CAP(SOURCE)=INF
      ITER=0
   99 LABEL(SOURCE)=1
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO  97
      ITER=ITER+1
      WRITE(6,225) ITER
C
C     INITIAL LABELING ITERATION
C
   97 I=1
      J=SOURCE
      M=0
C     IF ACTIVITY STARTS AT DESIGNATED ORIGIN, TRY TO LABEL.
C     OTHERWISE, CHANGE ORIGINS.
   14 IF (ORIG(I).NE.J) GO TO 13
      ITERM=TERM(I)
C     CHECK IF NODE ALREADY LABELED AND
C     CHECK IF ABAR(I,NK(I))=0.
      IF (LABEL(ITERM).NE.0.OR.ABAR(I,NK(I)).NE.0) GO TO 13
C     IF NODE NOT ALREADY LABELED AND ABAR(I,NK(I))=0,
C     PROCEDE WITH LABELING.
      LABEL(ITERM)=1
      ORIG2(ITERM)=J
      K1(ITERM)=NK(I)
      DIREC(ITERM)=1
      CAP(ITERM)=INF
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 403
      WRITE(6,201) ITERM,ORIG2(ITERM),K1(ITERM)
C     IF CAN REACH SINK, TERMINATE (IMPLIES INFEASIBLE)
  403 IF (ITERM.EQ.SINK) GO TO 15
      M=M+1
      KCUNT(M)=ITERM
C     IF EVERY PATH TESTED AND INFINITE FLOW NOT POSSIBLE,
C     GO ON TO LABELING PART(II).
   13 I=I+1
      IF (I.GT.NA) GO TO 11
      GO TO 14
C     CHANGE DESIGNATED ORIGINS.
   11 IF (J.EQ.SOURCE) P=1
      IF(P.GT.M) GO TO 16
C     IF ALL LABELED NODES HAVE BEEN SCANNED AND NO NEW NODES
C     HAVE BEEN LABELED, GO ON TO LABELING PART (II).
      J=KOUNT(P)
      P=P+1
```

```
      I=1
      GO TO 14
   15 IF(TEST3.GE.1) GO TO 404
      WRITE(6,202) LAMBDA
  404 GO TO 999
   16 IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 405
      WRITE(6,203)
C
C     NEXT LABELING PROCEDURE
C
  405 I=1
      J=SOURCE
C     AGAIN, CHECK ALL CONDITIONS FOR LABELING
C     IE. CHECK IF NODE IS ALREADY LABELED, IF ABAR(I,J)=0, AND
C     IF THE FLOW(I,J) IS LESS THAN ITS UPPER BOUND.
   20 IF (ORIG(I).NE.J) GO TO 24
      ITERM=TERM(I)
      KN=NK(I)
      DO 25 K=1,KN
      IF (K.EQ.KN) GO TO 27
      IF(LABEL(ITERM).NE.0.OR.ABAR(I,K).NE.0.OR.FLOW(I,K).GE.
     1(C(I,NK(I)-K)-C(I,NK(I)-K+1)))GO TO 25
      DIREC(ITERM)=I
C     CAPACITY IS MIN OF PREVIOUS FLOW AND THE EXCESS CAPACITY
      CAP(ITERM)=C(I,NK(I)-K)-C(I,NK(I)-K+1) - FLOW(I,K)
      GO TO 23
   27 IF(LABEL(ITERM).NE.0.OR.ABAR(I,K).NE.0.OR.FLOW(I,K).GE.INF)
     1 GO TO 25
C     IF THE NODE HAS NOT ALREADY BEEN LABELED, ABAR(I,J)=0, AND
C     THE FLOW IS LESS THAN ITS UPPER BOUND, PROCEDE WITH THE LABELING
C     OF THE NODE.
      DIREC(ITERM)=I
      CAP(ITERM)=INF
   23 LABEL(ITERM)=1
      ORIG2(ITERM)=J
      K1(ITERM)=K
      IF (CAP(ITERM).GT.CAP(ORIG(I)))  CAP(ITERM)=CAP(ORIG(I))
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 406
      WRITE(6,204) ITERM,ORIG2(ITERM),K1(ITERM),DIREC(ITERM),CAP(ITERM)
C     IF SINK LABELED, GO TO UPDATE PROCEDURE
  406 IF (ITERM.EQ.SINK) GO TO 21
      M=M+1
      KOUNT(M)=ITERM
C     CHECK IF ALL PATHS TRIED
   25 CONTINUE
      GO TO 19
   24 IF(TERM(I).NE.J) GO TO 19
      IORIG=ORIG(I)
      KN=NK(I)
      DO 26 K=1,KN
      IF(LABEL(IORIG).NE.0.OR.ABAR(I,K).NE.0.OR.FLOW(I,K).LE.0)
     2 GO TO 26
      DIREC(IORIG)=-I
      CAP(IORIG)=FLOW(I,K)
      LABEL(IORIG)=1
      ORIG2(IORIG)=J
      K1(IORIG)=K
      IF(CAP(IORIG).GT.CAP(TERM(I))) CAP(IORIG)=CAP(TERM(I))
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 402
      WRITE(6,204) IORIG,ORIG2(IORIG),K1(IORIG),DIREC(IORIG),CAP(IORIG)
  402 M=M+1
```

```
      KOUNT(M)=IORIG
   26 CONTINUE
   19 I=I+1
      IF (I.GT.NA) GO TO 18
      GO TO 20
   18 IF (J.EQ.SOURCE) P=1
      IF (P.GT.M) GO TO 22
      J=KOUNT(P)
      P=P+1
      I=1
      GO TO 20
C
C     NONBREAKTHROUGH HAS OCCURED. DELTAS ARE FOUND AND UPDATING
C     MADE IN THE XNODES AND XACTS.
C
   22 DELTA1=INF+1
      DELTA2=INF+1
      DO 4 I=1,NA
      KN=NK(I)
      IF (LABEL(ORIG(I)).EQ.1.AND.LABEL(TERM(I)).EQ.0) GO TO 1
C     A1 IS SET OF I LABELED AND J UNLABELED.
C     A2 IS SET OF I UNLABELED AND J LABELED.
      IF(LABEL(ORIG(I)).EQ.0.AND.LABEL(TERM(I)).EQ.1) GO TO 2
      GO TO 4
C     FINDING DELTA1'S.
    1 DO 3 J=1,KN
      IF (ABAR(I,J).GE.0) GO TO 3
      IF (-ABAR(I,J).LT.DELTA1) DELTA1=-ABAR(I,J)
    3 CONTINUE
      GO TO 4
C     FINDING DELTA2'S
    2 DO 5 J=1,KN
      IF(ABAR(I,J).LE.0) GO TO 4
      IF (ABAR(I,J).LT.DELTA2) DELTA2= ABAR(I,J)
    5 CONTINUE
    4 CONTINUE
C     DEL=MIN(DELTA1,DELTA2)
      DEL=DELTA1
      IF (DELTA2.LT.DEL) DEL=DELTA2
      LAMBDA=LAMBDA-DEL
      KKKK=KKKK+1
      FILEO(KKKK)=LAMBDA
C     UPDATING THE XNODES.
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 407
      WRITE(6,206) LAMBDA
  407 IF (TEST3.EQ.2) GO TO 721
      DELTA= LAMBDA + DEL
      WRITE(6,209)          DEL,DELTA,LAMBDA,DELTA
  721 IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 408
      WRITE(6,207)
  408 DO 80 I=1,LNODEN
      INODE=XNODE(I)
      IF(LABEL(I).EQ.0) GO TO 81
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 409
      WRITE(6,210) I,INODE
  409 XNODE(I)=INODE
      GO TO 80
   81 IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 410
      WRITE(6,211) I,INODE
  410 XNODE(I)=INODE-DEL
   80 CONTINUE
```

```
C     UPDATING THE XACTS (3 PARTS)
      IF (TEST3.EQ.2) GO TO 722
      WRITE(6,212) DELTA
  722 PCOST=0.
      DO 82 I=1,NA
      IP(I)=0
      PCOST1=0.
      NKM1=NK(I)-1
      IACT=TIME(I,NK(I))
      IORIG=ORIG(I)
      ITERM=TERM(I)
      IDIFF=XNODE(ITERM)-XNODE(IORIG)
      XDIFF(I)=-IDIFF
      IF (IDIFF.GE.IACT) GO TO 86
      XACT(I)=IDIFF
      DO 550 K=1,NKM1
      IF(K.NE.1) GO TO 43
      XIJ=XACT(I)
      IF(XIJ.GT.TIME(I,2)) XIJ=TIME(I,2)
      FLAG1=0
      GO TO 42
   43 XIJ=XACT(I)-TIME(I,K)
      IF(XIJ.LT.0) GO TO 552
      IF(XIJ.GT.(TIME(I,K+1)-TIME(I,K))) XIJ=TIME(I,K+1)-TIME(I,K)
      FLAG1=0
      GO TO 42
  552 FLAG1=1
      FLAG2=K-1
      GO TO 553
   42 PCOST1=PCOST1+C(I,K)*XIJ
  550 CONTINUE
  553 KCOST=COST(I,1)+C(I,1)*TIME(I,1)
      ACOST=KCOST-PCOST1
      PCOST=PCOST+ACOST
      IF (LABEL(IORIG)-LABEL(ITERM)) 83,84,85
   83 IDIFF=IDIFF-DEL
      FILE6(1,I)=1
      FILE6(2,I)=IDIFF
      IF (TEST3.EQ.2) GO TO 82
      IF(FLAG1.EQ.1) GO TO 59
      ACOST=ACOST + C(I,NKM1)*DEL
      IP(I)=1
      WRITE(6,214)NUMS(I),IDIFF,ACOST,C(I,NKM1)
      GO TO 82
   59 ACOST=ACOST+C(I,FLAG2)*DEL
      IP(I)=1
      WRITE(6,214)NUMS(I),IDIFF,ACOST,C(I,FLAG2)
      GO TO 82
   84 FILE6(1,I)=0
      FILE6(2,I)=XACT(I)
      IF(TEST3.EQ.2)GO TO 82
      WRITE(6,216)NUMS(I),XACT(I),ACOST
      GO TO 82
   85 IDIFF=IDIFF+DEL
      FILE6(1,I)=-1
      FILE6(2,I)=IDIFF
      IF(TEST3.EQ.2)GO TO 82
      IF(FLAG1.EQ.1) GO TO 58
      ACOST=ACOST - C(I,NKM1)*DEL
      IP(I)=2
      WRITE(6,213)NUMS(I),IDIFF,ACOST,C(I,NKM1)
```

```
      GO TO 82
   58 ACOST=ACOST-C(I,FLAG2)*DEL
      IP(I)=2
      WRITE(6,213)NUMS(I),IDIFF,ACOST,C(I,FLAG2)
      GO TO 82
   86 XACT(I)=IACT
      DO 551 K=1,NKM1
      IF(K.NE.1) GO TO 45
      XIJ=XACT(I)
      IF(XIJ.GT.TIME(I,2)) XIJ=TIME(I,2)
      GO TO 46
   45 XIJ=XACT(I)-TIME(I,K)
      IF(XIJ.LT.0) XIJ=0
      IF(XIJ.GT.(TIME(I,K+1)-TIME(I,K))) XIJ=TIME(I,K+1)-TIME(I,K)
   46 PCOST1=PCOST1+C(I,K)*XIJ
  551 CONTINUE
      KCOST=COST(I,1)+C(I,1)*TIME(I,1)
      ACOST=KCOST-PCOST1
      PCOST=PCOST+ACOST
      FILE6(1,I)=0
      FILE6(2,I)=XACT(I)
      IF (TEST3.EQ.2) GO TO 82
      WRITE(6,216)NUMS(I),XACT(I),ACOST
   82 CONTINUE
      WRITE(F6)((FILE6(I,J),I=1,2),J=1,NA)
      IF (TEST3.EQ.2) GO TO 723
      PCOST1=PNEW
      PNEW=(PCOST-PNEW)/DEL
      WRITE(6,224) PCOST1,PNEW
  723 PNEW=PCOST
      IF (TEST3.NE.0.AND.LAMBDA.LE.LINPUT) GO TO 703
C     RESET LABELS TO 0 AND REFIGURE ABARS.
C     THEN START OVER.
      DO 87 I=1,LNODEN
      LABEL(I)=0
   87 CONTINUE
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 420
      WRITE(6,226) (J,J=1,11)
  420 DO 88 I=1,NA
      NKM1=NK(I)-1
      DO 500 K=1,NKM1
      J=NKM1+2-K
  500 ABAR(I,K)=TIME(I,J)+XDIFF(I)
      ABAR(I,NK(I))=TIME(I,      1)+XDIFF(I)
      IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO  88
      NK1=NK(I)
      WRITE(6,227)NUMS(I),(ABAR(I,J),J=1,NK1)
   88 CONTINUE
      IF (LAMBDA.LT.LMIN) GO TO 998
      GO TO 99
C
C     UPDATE THE FLOW AFTER BREAKTHROUGH.
C
   21 IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO  34
      WRITE(6,205)
   34 FLOW(I,KI(ITERM))=FLOW(I,KI(ITERM))+CAP(SINK)
C     IF DIREC =0 THEN CAP ADDED TO FLOW.
C     IF DIREC =1 THEN CAP IS SUBTRACTED.
   30 ITERM=ORIG2(ITERM)
C     CHECK IF BACK AT SOURCE.
      IF(ITERM.EQ.SOURCE) GO TO 33
```

```
      I=DIREC(ITERM)
      I=IABS(I)
      IF (DIREC(ITERM).GT.0) GO TO 34
      FLOW(I,K1(ITERM))=FLOW(I,K1(ITERM))-CAP(SINK)
      GO TO 30
C     RELABEL AND START OVER.
   33 IF(TEST2.EQ.1.OR.TEST3.GE.1) GO TO 415
      DO 560 I=1,NA
      NK1=NK(I)
      DO 560 K=1,NK1
  560 WRITE(6,220)NUMS(I),K,FLOW(I,K)
  415 DO 98 I=1,LNODEN
      LABEL(I)=0
   98 CONTINUE
      GO TO 99
C     PROGRAM TERMINATES WHEN EVENTUALLY AN INFINITE FLOW IS ACHIEVED
C     FROM THE SOURCE TO THE SINK,  OR WHEN THE VALUE OF LAMBDA DROPS
C     BELOW THE MINIMUM LENGTH OF THE NETWORK.
  998 IF(TEST3.NE.0) GO TO 999
      WRITE(6,202) LAMBDA
      GO TO 999
  705 WRITE(6,233) LINPUT,LMIN
      GO TO 999
  704 WRITE(6,236) LINPUT,LMAX
      WRITE(6,238) LINPUT
      D=0
      DO 60 I=1,NA
   60 IP(I)=0
      GO TO 707
  703 WRITE(6,234) LINPUT
  706 WRITE(6,238) LINPUT
      D=LINPUT-LAMBDA
  707 PCOST=0.
      DO 57 I=1,NA
      IF(IP(I).EQ.1.AND.D.GT.0) XACT(I)=XACT(I)-D
      IF(IP(I).EQ.2.AND.D.GT.0) XACT(I)=XACT(I)+D
      PCOST1=0.
      NKM1=NK(I)-1
      DO 51  K=1,NKM1
      IF(K.NE.1) GO TO 52
      XIJ=XACT(I)
      IF(XIJ.GT.TIME(I,2)) XIJ=TIME(I,2)
      GO TO 53
   52 XIJ=XACT(I)-TIME(I,K)
      IF(XIJ.LT.0) XIJ=0
      IF(XIJ.GT.(TIME(I,K+1)-TIME(I,K))) XIJ=TIME(I,K+1)-TIME(I,K)
   53 PCOST1=PCOST1+C(I,K)*XIJ
   51 CONTINUE
      KCOST=COST(I,1)+C(I,1)*TIME(I,1)
      ACOST=KCOST-PCOST1
      WRITE(6,216)NUMS(I),XACT(I),ACOST
   57 PCOST=PCOST+ACOST
      WRITE(6,239) PCOST
  999 WRITE(6,228)
      READ(F0)
      WRITE(F0)KKKK,(FILE0(I),I=1,KKKK)
      ENDFILE F6
      ENDFILE F0
      WRITE(6,228)
      STOP
  100 FORMAT(I4,1X,I4,1X,I1,1X,I1,1X,I4,1X,I4,1X,I1)
```

```
150 FORMAT('-','THE NUMBER OF NODES IS ',I4,'.',/,1X,'THE NUMBER OF AC
   1TIVITIES IS ',I4,'.',/,1X,'THE SOURCE NODE IS NUMBERED ',I4,' AND
   2THE SINK NODE IS NUMBERED ',I4,'.',/,'-',' ** NODES: **')
151 FORMAT('0',16X,'K',7X,9(3X,I4,5X))
152 FORMAT('-',' ** ACTIVITIES: **',//,6X,'I',7X,'XACT',6X,'ORIG',3X,
   1'TERM',4X,'J',6X,'TIME',9X,'COST',14X,'C',13X,'ABAR')
153 FORMAT(' ',3X,I4,3X,I10,3X,I4,3X,I4,(T39,I2,3X,I10,3X,I10,3X,
   1E16.5,3X,I10))
154 FORMAT('-','THE ENTIRE PROJECT COST CURVE WILL BE DETERMINED.')
155 FORMAT('-','THE OPTIMAL ACTIVITY COMPLETION TIMES FOR A SPECIFIED
   1PROJECT DEADLINE TIME = ',I10,' ARE GOING TO BE DETERMINED.')
156 FORMAT(' ',T39,I2,3X,I10,3X,I10,22X,I10)
157 FORMAT('0',                              4X,'INITIAL XNODE(K)',3X,
   19(I10,2X))
200 FORMAT('0','LAMBDA = PROJECT COMPLETION TIME',//,
   1         1X,'THE STARTING VALUE OF LAMBDA IS ',I10,'.',//
   2         ,1X,'THE CORRESPONDING TOTAL PROJECT COST IS ',E16.5,'.')
201 FORMAT('0','THE NODE ',I4,' HAS THE LABEL (',I4,',',I4,',0,INF).')
202 FORMAT('0',////,30X,'* * * * * *',/////,1X,
   1         'THE SINK WAS REACHED WITH INFINITE CAPACITY IMPLYING A
   1N INFEASIBLE SOLUTION TO THE PRIMAL PROBLEM ',/,20X,'IF LAMBDA DRO
   2PS BELOW ITS CURRENT VALUE, ',I10,'.')
203 FORMAT('-','THE SINK HAS NOT BEEN REACHED WITH INFINITE CAPACITY -
   1 CONTINUE WITH THE LABELING PROCESS.',/  ,1X,'THE NODES THAT HAVE
   2BEEN LABELED WILL RETAIN THAT LABEL FOR THE REMAINDER OF THE ITERA
   3TION.')
204 FORMAT('0','THE NODE ',I4,' HAS THE LABEL (',I4,',',I4,',',I4,',',
   1E16.5,').')
205 FORMAT('-','BREAKTHROUGH:  UPDATE THE DUAL VARIABLES.'
   1,///,1X,'  ACTIVITY #: I',3X,'J',9X,'NEW FLOW: F(I,J)')
206 FORMAT('-','NONBREAKTHROUGH:  UPDATE THE PRIMAL VARIABLES;',/,1X,
   1'I.E.  DETERMINE OPTIMAL ACTIVITY TIMES FOR LAMBDA = ',I10,'.')
207 FORMAT(' ','  NODE #: K',5X,'NEW VALUE: XNODE(K)')
209 FORMAT('-','DELTA (REPRESENTED BY "D") RANGES FROM 0 TO'
   1 ,I4,'.',/,1X,'LAMBDA RANGES FROM',I10,' TO',I10,'.',
   2              /,1X,'THE MINIMUM COST PROJECT SCHEDULE FOR PRO
   3JECT DEADLINE = ',I10,'-D:')
210 FORMAT(' ',7X,I4,12X,I10)
211 FORMAT(' ',7X,I4,12X,I10,'-D')
212 FORMAT('-','PROJECT COMPLETION TIME = ',I10,'-D.',//,1X,
   1          '  ACTIVITY #: I',3X,'NEW VALUE: XACT(I)',9X,'ACTIVITY CO
   2ST')
213 FORMAT(' ',5X,I4,12X,I10,'-D',9X,E16.5,' + (',E13.5,'*D)')
214 FORMAT(' ',5X,I4,12X,I10,'+D',9X,E16.5,' + (',E13.5,'*D)')
216 FORMAT(' ',5X,I4,12X,I10,11X,E16.5)
220 FORMAT(' ',12X,I4,2X,I2,7X,E16.5)
224 FORMAT('0','THE CURRENT VALUE OF THE PROJECT COST IS ',E16.5,
   1' + (',E13.5,'*D).')
225 FORMAT('-','***        ITERATION NUMBER',I6,'       ***')
226 FORMAT('-','NEW VALUES OF ABAR FOR J=1,2,...,NK(I)',//,6X,'I',3X,
   1'J:',11(5X,I2,3X))
227 FORMAT(' ',2X,I4,7X,11(I8,2X))
228 FORMAT(1H1)
230 FORMAT(I4,1X,I4,1X,I2)
231 FORMAT(8I10)
232 FORMAT(I10)
233 FORMAT('-','THE SPECIFIED VALUE OF LAMBDA,',I10,',IS LESS THAN THE
   1 MINIMUM VALUE,',I10,',IMPLYING AN INFEASIBLE SOLUTION.',//,1X,
   2'THE PROBLEM WILL NOT BE WORKED.')
234 FORMAT('1','THE SPECIFIED VALUE OF LAMBDA,',I10,',HAS BEEN REACHED
   1.')
```

```
  235 FORMAT('0',   'THE SOURCE HAS A VALUE OF ZERO AND IS ASSIGNED THE
     3LABEL (-,-,-,INF).',// )
  236 FORMAT('-','THE SPECIFIED VALUE OF LAMBDA, ',I10,',IS GREATER THAN
     1 OR EQUAL TO THE MAXIMUM VALUE, ',I10,';',//,
     1                                          1X,'THEREFORE, THE ORIGINAL
     2XNODE(K)''S AND XACT(I)''S ARE OPTIMAL.')
  237 FORMAT(' ','** WARNING:  ACTIVITY NUMBER ',I4,' HAS A NON-CONVEX C
     1OST FUNCTION;',/,12X,'IE. THE C(',I4,',M)''S ARE NOT NON-INCREASIN
     2G.')
  238 FORMAT('-','FOR PROJECT COMPLETION TIME = ',I10,', THE OPTIMAL SOL
     1UTION IS:',//,1X,
     1         ' ACTIVITY #: I',3X,'NEW VALUE: XACT(I)',9X,'ACTIVITY CO
     2ST')
  239 FORMAT('-','THE CORRESPONDING PROJECT COST IS ',E16.5,'.')
      END
      SUBROUTINE ORDER
C
C          THIS SUBROUTINE DETERMINES THE ORDER IN WHICH TO CONSIDER
C          THE ACTIVITIES FOR THE CALCULATION OF THE CRITICAL PATH TIME
C
      IMPLICIT INTEGER*2(A-Z)
      INTEGER*4 NA,NN,ORIG,TERM,NK,LNODEN
      COMMON ORIG,TERM,NK,NA,LNODEN,TIME,CTIME,XNODE,AORD,LMIN,LMAX,
     *        TEST1
      DIMENSION ORIG(1000),TERM(1000),AORD(1000),CTIME(1000),
     *XNODE(9999),ND(9999),NDD(9999),TIME(1000, 6),NK(1000)
      N=LNODEN
      M=NA
      NDD(1)=1
      DO 5 I=2,N
5     NDD(I)=0
      DO 6 I=1,M
6     AORD(I)=0
      K=0
      MP=M+1
      DO 1 II=1,MP
      DO 20 I=1,N
20    ND(I)=NDD(I)
      III=0
      IP=II+1
      DO 2 J=1,M
      IF(ND(ORIG(J)).NE.II) GO TO 2
      NDD(TERM(J))=IP
      III=1
      IF(K.EQ.0) GO TO 14
      DO 10 L=1,K
      IF(AORD(L).EQ.J) GO TO 11
10    CONTINUE
14    K=K+1
      GO TO 13
11    IF(L.EQ.K) GO TO 2
      KM=K-1
      DO 12 LL=L,KM
12    AORD(LL)=AORD(LL+1)
13    AORD(K)=J
2     CONTINUE
      IF(III.EQ.0) GO TO 3
1     CONTINUE
3     CONTINUE
      DO 30 I=1,NA
30    CTIME(I)=TIME(I,1)
```

AD-A060 568 TEXAS A AND M UNIV COLLEGE STATION INST OF STATISTICS F/G 9/2
A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJEC--ETC(U)
AUG 78 T C BAKER, R L SIELKEN N00014-78-C-0426
UNCLASSIFIED THEMIS-TR-57 NL

3 OF 4
AD
A060568

```
      LMIN=CPTIME(CPATHT)
      DO 31 I=1,NA
      NK1=NK(I)
31    CTIME(I)=TIME(I,NK1)
      LMAX=CPTIME(CPATHT)
      RETURN
      END
      FUNCTION CPTIME(CPATHT)
C
C          DETERMINE THE CRITICAL PATH TIME: CPTIME
C          XNODE(I) = EARLIEST TIME THAT AN ACTIVITY BEGINNING AT NODE I
C                   CAN COMMENCE
C
      IMPLICIT INTEGER*2(A-Z)
      INTEGER*4 NA,NN,ORIG,TERM,NK,LNODEN
      COMMON ORIG,TERM,NK,NA,LNODEN,TIME,CTIME,XNODE,AORD,LMIN,LMAX,
     *        TEST1
      DIMENSION ORIG(1000),TERM(1000),AORD(1000),CTIME(1000),
     *XNODE(9999),ND(9999),NDD(9999),TIME(1000, 6),NK(1000)
      DO 1 I=1,LNODEN
1     XNODE(I)=0
      DO 2 II=1,NA
      I=AORD(II)
2     IF(XNODE(ORIG(I))+CTIME(I).GT.XNODE(TERM(I)))
     1   XNODE(TERM(I))=XNODE(ORIG(I))+CTIME(I)
      CPTIME=XNODE(TERM(II))
      II=AORD(NA)
      RETURN
      END
```

3. DSR

```
C      DETERMINISTIC SCHEDULE RESOLUTION PROGRAM
C
C          THIS PROGRAM ASCERTAINS THE DISTRIBUTIONAL PARAMETERS FOR EACH
C          ACTIVITY IN THE PROJECT NETWORK THAT CORRESPOND TO THE MINIMUM
C          COST PROJECT SCHEDULE WHICH YIELDS A DETERMINISTIC COMPLETION
C          TIME OF PD.   THIS IS DONE BY REFERRING TO THE OUTPUT FROM THE
C          DETERMINISTIC PROJECT SCHEDULER WHICH HAS BEEN STORED ON DISK.
C
C          THE ACTIVITY DISTRIBUTIONS ARE USED BY SIMPLIFICATION AND
C          INDIRECTLY BY SUBNETWORK ANALYSIS.
C
C          FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C          MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                        PROJECT NETWORK
C          BKMAX = THE MAXIMUM NUMBER OF BREAK POINTS ALLOWED IN THE
C                        ENTIRE PROJECT'S TIME-COST CURVE
C          NCTMAX = THE MAXIMUM NUMBER OF COMPLETION TIMES AND COSTS FOR
C                        EACH ACTIVITY
C          CURRENTLY, MMAX=1000; BKMAX=3000; NCTMAX=6
C
C       INTEGER*2 BREAK(BKMAX),FILE6(2*MMAX),TYME(NCTMAX)
C       DIMENSION TIME(4),NCT(MMAX),FILE0(4*MMAX+5),NUMS(MMAX)
C       EQUIVALENCE (FILE0(3*MMAX+6),NCT(1))
C
        INTEGER*2 NBREAK,BREAK(3000),FILE6(2000),TYME(6)
        REAL*8 AA,XM,BB,ALPHA,BETA
        DIMENSION TIME(4),NCT(1000),FILE0(4005),NUMS(1000)
        EQUIVALENCE (FILE0(1),NA),(FILE0(6),NUMS(1)),(FILE0(3006),NCT(1))
        INTEGER F0/8/,F1/9/,F2/10/,        F4/12/,        F6/14/
        INTEGER NORM(5)/'NORM','AL',3*' '/,BET(5)/'BETA',4*' '/
        INTEGER RECT(5)/'RECT','ANGU','LAR(','UNIF','ORM)'/
        INTEGER FIXED(5)/'FIXE','D(CO','NSTA','NT)',' '/
C
C       FOR EACH ACTIVITY--
C      USES ORIGINAL TIME VECTOR AND RESULTS OF THE DETERMINISTIC PROJECT
C      SCHEDULER TO CREATE A NEW TIME VECTOR.  THIS DETERMINES THE ACTIV-
C      ITY DISTRIBUTIONS TO BE USED BY SIMPLIFICATION AND INDIRECTLY BY
C      SUBNETWORK ANALYSIS.
C
        WRITE(6,1500)
 1500   FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE DETE
       *RMINISTIC SCHEDULE RESOLUTION PROGRAM: DSR'
       *//1X,132('*')/1X,132('*'))
        REWIND F0
        REWIND F1
        REWIND F2
        REWIND F4
        REWIND F6
        READ(F0)FILE0
        READ(F1)
        READ(F4)NCYC,PD,NFLAG
        WRITE(6,2000)PD
 2000   FORMAT(1H0,/'0THE MINIMUM COST PROJECT SCHEDULE FOR A TARGET TIME
       *OF ',F12.5,' IS AS FOLLOWS:'//'0ACTIVITY:',8X,            'MEA
       *N',8X,'DISTRIBUTION',14X,'PARAMETERS'/)
        READ(F0)
        READ(F0)NBREAK,(BREAK(I),I=1,NBREAK)
        REWIND F6
        DO 1 I=1,NBREAK
        J=I
```

```
      IF(PD.GE.BREAK(J))GO TO 2
1     CONTINUE
      NBB=BREAK(NBREAK)
      WRITE(6,100)NBB,PD
100   FORMAT(1H0,'THE SMALLEST POSSIBLE PROJECT COMPLETION TIME BASED ON
     * NON-RANDOM INDIVIDUAL ACTIVITY COMPLETION TIMES IS ',I10,'.'/
     *'0THE OPTIMAL ACTIVITY COMPLETION TIMES FOR A TARGET TIM',
     *E OF ',E14.5,' HAVE BEEN REQUESTED.')
      IF(NFLAG.EQ.1)GO TO 300
      PD=NBB
      WRITE(6,101)PD
101   FORMAT(' THE OPTIMAL ACTIVITY COMPLETION TIMES FOR A TARGET',
     *  ' TIME OF',E14.5,' WILL BE SUPPLIED.')
      NFLAG=1
      REWIND F4
      WRITE(F4)NCYC,PD,NFLAG
      D=0.0
      I=NBREAK-1
      GO TO 200
300   WRITE(6,102)
102   FORMAT('0THIS IS THE SECOND CONSECUTIVE REQUEST FOR AN INFEASIBLE
     *PROJECT DEADLINE TIME.'/' A FATAL ERROR WILL BE COMMITTED SO THAT
     *ALL REMAINING JOB STEPS WILL BE SKIPPED.')
      CALL ERRSET(251,1,-1,1)
      XW=DSQRT(-1.D0)
      GO TO 400
2     NFLAG=0
      D=0.D0
      I=J-1
      IF(I.EQ.0)GO TO 4
      D=BREAK(I)-PD
200   DO 3 J=1,I
3     READ(F6)
4     N1=2*NA
      READ(F6)(FILE6(I),I=1,N1)
      DO 700 JJ=1,NA
      J=JJ*2
      EMA=FILE6(J)+D*FILE6(J-1)
      NNCT=NCT(JJ)
      NNCT1=NNCT-1
      READ(F1)(TYME(J),J=1,NNCT)
      DO 709 J=1,NNCT
      IF(EMA.GE.TYME(J))GO TO 709
      JK=J-1
      GO TO 708
709   CONTINUE
      JK=NNCT1
708   DO 707 J=1,NNCT1
      IF(J-JK)706,705,706
705   READ(F1)IDIST,TIME
      GO TO 707
706   READ(F1)
707   CONTINUE
      MFIX=IDIST+2
      GO TO (710,711,712,715),MFIX
710   CONTINUE
C
C     BETA DISTRIBUTION
C
      IF(TIME(4)   .GT.-1.D0)GO TO 7100
C
```

```
C     CONVERT THE MOST LIKELY PARAMETER, TIME(2), TO ALPHA AND BETA
C
      AA = TIME(1)
      XM=TIME(2)
      BB = TIME(3)
      CALL ALBET(AA,XM,BB,ALPHA,BETA)
      TIME(2)    = ALPHA
      TIME(4)    = BETA
7100  CONTINUE
      A     = TIME(2)    + 1.
      B     = TIME(4)    + 1.
      XMIN = TIME(1)
      XMAX = TIME(3)
C
C     COMPUTE NEW ALPHA,BETA,MIN TIME, AND MAX TIME
C
      UA = (XMAX-XMIN)*(A/(A+B)) + XMIN
      C  = (UA-XMIN)/(XMAX-UA)
      V = A*B/((A+B)**2*(A+B+1.))
      TIME(1)    = XMIN*EMA/UA
      TIME(4)    = (C/(V*(C+1)*(C+1))-1.)/(C+1)-1.
      TIME(3)    = XMAX*EMA/UA
      TIME(2)    = C*(TIME(4)    +1.)-1.
      WRITE(6,2101)NUMS(JJ),EMA,BET, TIME(1),TIME(3),TIME(2),TIME(4)
2101  FORMAT(I6,6X,F12.5,6X,5A4,6X,'MIN=',F13.5,', MAX=',F15.5,', ALPHA=
     *',F12.5,', BETA=',F12.5)
      GO TO 713
711   CONTINUE
C
C     NORMAL DISTRIBUTION
C
      IF (TIME(4)    .EQ.1.) GO TO 714
      TIME(3)    =(TIME(3)    -TIME(1)    )/6.
      TIME(4)    = 1.
      TIME(1)    =0.
714   TIME(3)    = EMA/TIME(2)    *TIME(3)
      TIME(2)    = EMA
      WRITE(6,2201)NUMS(JJ),EMA,NORM,TIME(2),TIME(3)
2201  FORMAT(I6,6X,F12.5,6X,5A4,6X,'MEAN=',F12.5,', ST.DEV=',F12.5)
      GO TO 713
712   CONTINUE
C
C     UNIFORM DISTRIBUTION
C
      TDUM = 2.*EMA*TIME(1)    /(TIME(1)    +TIME(3)    )
      TIME(3)    = 2.*EMA*TIME(3)    /(TIME(1)    +TIME(3)    )
      TIME(1)    = TDUM
      WRITE(6,2301)NUMS(JJ),EMA,RECT,TIME(1),TIME(3)
2301  FORMAT(I6,6X,F12.5,6X,5A4,6X,'MIN=',F13.5,', MAX=',F15.5)
      GO TO 713
715   WRITE(6,2401)NUMS(JJ),EMA,FIXED,TIME(1)
2401  FORMAT(I6,6X,F12.5,6X,5A4,6X,'TIME=',F12.5)
713   CONTINUE
      WRITE(F2)IDIST,TIME
700   CONTINUE
400   STOP
      END
      SUBROUTINE ALBET (AA,XM,BB,ALPHA,BETA)
C SUBROUTINE TO CALCULATE THE ALPHA AND BETA VALUES OF THE BETA DENSITY.
C N#6.SEE PAPER BY A.W.WORTHAM AND R.E.COLE,INDUSTRIAL ENGINEERING DEPT.
C
```

```
C     GIVEN AA, BB, AND XM THE VALUES OF ALPHA AND BETA ARE DETERMINED
C     FROM A AND THE STANDARD APPROXIMATION B.
C
C      CALLING ARGUMENTS.
C
C           1. AA IS THE MINIMUM TIME ESTIMATE.
C           2.  XM IS THE MOST LIKLEY TIME ESTIMATE.
C           3.  BB IS THE MAXIMUM TIME ESTIMATE.
C           4. ALPHA AND BETA ARE THE VALUES CALCULATED
C TERPOL IS A ARITHMETIC STATEMENT FUNCTION TO DO LINEAR INTERPOLATION.
C IF X IS THE VALUE REQUIRED WHICH LIES BETWEEN THE VALUES  OF X1 AND X2
C IN A TABLEX X2 IS THE LARGER VALUE< Y1 AND Y2 THE CORRESPONDING VALUES
C IN THE TABLE.XY2 GREATER THAN Y GREATER THAN Y1<.
      IMPLICIT REAL*8(A-H,O-Z)
      TERPOL(X1,X2,Y,Y1,Y2)=X1+((X2-X1)*(Y-Y1)/(Y2-Y1))
      KK=-1
      K=0
      KA=AA*10.D0 + 0.1D0
      KM=XM*10.D0 + 0.1D0
      KB=BB*10.D0 + 0.1D0
      KB=KB-KM
      KA=KM-KA
      IF(XM.EQ.BB) GOTO 22
      IF(XM.EQ.AA) GOTO 21
      GOTO 23
   21 ALPHA=0.D0
      BETA=2.8724D0
      GOTO 20
   22 ALPHA=2.8724D0
      BETA=0.D0
      GOTO 20
   23 IF(KA-KB) 5,7,5
 5    S=(XM-AA)/(BB-XM)
      IF(S-1.D0) 6,7,8
    7 ALPHA=3.D0
      BETA=3.D0
      GO TO 20
    6 S=1.D0/S
      K=-1
    8 Y=6.D0
      DO 10 I=1,600
      Y=Y-1.D-2
      B=Y*36.D0
      C=(Y**3)+(7.D0*(Y**2))-20.D0*Y -24.D0
      R=(B**2)-(144.D0*C)
      ROOT=DSQRT(R)
      ALF=(B+ROOT)/72.D0
      BET=(B-ROOT)/72.D0
      SS=ALF/BET
      IF (SS- S) 10,11,14
 14   IF (KK)15,16,11
 15   KK=KK+1
      S1=SS
      X1=ALF
      Z1=BET
      GOTO 10
 10   CONTINUE
      WRITE  (6,18)
 18   FORMAT(1H1, 32H ERROR,NO VALUE OF ALPHA BETA     )
      CALL EXIT
 16   KK=KK+1
```

```
      X2=ALF
      Z2=BET
      S2=SS
      ALF=TERPOL(X1,X2,S,S1,S2)
      BET=TERPOL(Z1,Z2,S,S1,S2)
 11   IF (K) 12,13,13
 12   BETA =ALF
      ALPHA = BET
      GO TO 20
 13   ALPHA = ALF
      BETA = BET
 20   RETURN
      END
```

4. SIMP

```
C      SIMPLIFICATION PROGRAM
C
C         THIS PROGRAM DETERMINES THE SIMPLIFIED PROJECT NETWORK BY
C         REPLACING CERTAIN CONFIGURATIONS OF ACTIVITIES WITH SINGLE
C         ACTIVITIES WHOSE DISTRIBUTIONS CAN BE DETERMINED TO MATCH THE
C         DISTRIBUTIONS OF THE CONFIGURATIONS OF ACTIVITIES REPLACED.
C
C         FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS. LET
C         MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                    PROJECT NETWORK
C         CURRENTLY. MMAX=1000
C
C      INTEGER FILE3(3*NA),S(NA,4)
C
       IMPLICIT REAL*8(A-H,O-R,T-Z)
       INTEGER F0/8/,              F3/11/,                    F7/15/
       INTEGER            FILE3(3000),S(1000,4),LIST(24),JP(20)
       EQUIVALENCE (FILE3(1),S(1))
       EQUIVALENCE (ISBR,LIST(1)),(I,LIST(2)),(JP(1),LIST(3))
       WRITE(6,9000)
9000   FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE SIMP
      *LIFICATION PROGRAM: SIMP'/1H0,132('*')/1X,132('*'))
       REWIND F0
       NRED=0
       READ(F0)NACT,NODES,NSRCE,NSINK,LNODEN,S
       READ(F0)JMAT
       NK=NACT
       REWIND F3
       READ(F3)
       DO 99 I=1,NK
99     S(I,4)=1
C
C
C         BUILD FILE7 FOR THE MODIFIED SIMPLIFICATION PROGRAM
C
C      START OF SEARCH FOR REDUCIBLE NETWORKS
C
  100 DO 190 I=1,NK
       IF(S(I,4).LE.0)  GO TO 190
       KP=0
       DO 121 J=1,NK
       IF(I.NE.J.AND.S(J,4).GT.0.AND.S(I,2).EQ.S(J,2).AND.S(I,3).EQ.S(J,3
     1)) GO TO 122
       GO TO 121
  122 KP=KP+1
       JP(KP)=J
       IF(KP.GE.20) GO TO 200
  121 CONTINUE
       IF(KP)120,120,200
  120 DO 130 J=1,NK
       IF(I.NE.J.AND.S(J,4).GT.0.AND.S(I,3).EQ.S(J,2)) GO TO 125
  130 CONTINUE
       GO TO 190
  125 J1=J
       JK=J1+1
       IF(JK-NK)127,127,141
  127 DO 140 J=JK,NK
       IF(I.NE.J.AND.S(J,4).GT.0.AND.S(J,2).EQ.S(J1,2)) GO TO 160
  140 CONTINUE
  141 DO 150 J=1,NK
```

```
      IF(I.NE.J.AND.J1.NE.J.AND.S(J,4).GT.0.AND.S(J,3).EQ.S(I,3)) GO TO
     1190
  150 CONTINUE
      GO TO 300
  160 J2=J
      K3=-1
      K4=-1
      K5=-1
  161 DO 165 J=1,NK
      IF(I.EQ.J.OR.J1.EQ.J.OR.J2.EQ.J.OR.S(J,4).LE.0) GO TO 165
      IF(S(J,3).EQ.S(I,3))GO TO 190
  165 CONTINUE
C
C     CHECK FOR FIRST ORDER CROSSED NETWORKS
C
   49 KK=0
   50 DO 60 J=1, NK
      IF(I.EQ.J.OR.J.EQ.J1.OR.J.EQ.J2.OR.S(J,4).LE.0) GO TO 60
      IF(S(J,2).EQ.S(I,2).AND.S(J,3).EQ.S(J2,3)) GO TO 51
      GO TO 60
   51 J3=J
   52 K3=K3+1
      IF(K3) 60,60,190
   60 CONTINUE
      IF (K3) 61,62,190
   61 IF(KK) 53,53,62
   53 L=J1
      J1=J2
      J2=L
      KK=1
      GO TO 50
   62 DO 170 J=1,NK
      IF(I.EQ.J.OR.J.EQ.J1.OR.J.EQ.J2.OR.S(J,4).LE.0) GO TO 170
      IF(S(J,2).EQ.S(J2,3).AND.S(J,3).EQ.S(J1,3)) GO TO 167
      IF(S(J,2).EQ.S(J1,3).AND.S(J,3).EQ.S(J2,3)) GO TO 168
      GO TO 170
  167 J4=J
      K4=K4+1
      IF(K4)170,170,190
  168 J5=J
      K5=K5+1
      IF(K5)170,170,190
  170 CONTINUE
      IF(K3.EQ.0.AND.K4.EQ.0.AND.K5.EQ.-1)GO TO 180
      IF(K3.EQ.0.AND.K4.EQ.-1.AND.K5.EQ.0) GO TO 171
      GO TO 190
  171 K6=-1
      K7=-1
      DO 175 J=1,NK
      IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J3.OR.J.EQ.J5.OR.S(J,4).LE.0.OR.J.EQ.
     1J2)  GO TO 175
      IF(S(J,2).EQ.S(J1,3)) GO TO 172
      IF(S(J,2).EQ.S(J3,3)) GO TO 173
      IF(S(J,3).EQ.S(J1,3).OR.S(J,3).EQ.S(J3,3).OR.S(J,2).EQ.S(I,3)) GO
     1TO 190
      GO TO 175
  172 J6=J
      K6=K6+1
      IF(K6)175,175,190
  173 J7=J
      K7=K7+1
```

```
      IF(K7)175,175,190
  175 CONTINUE
      IF(K6.EQ.0.AND.K7.EQ.0) GO TO 176
      GO TO 190
  176 IF(S(J6,3).EQ.S(J7,3)) GO TO 600
      GO TO 190
  180 K8=-1
      K9=-1
      K12=-1
      K13=-1
      DO 185 J=1,NK
      IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J2.OR.J.EQ.J3.OR.J.EQ.J4.OR.S(J,4).LE
     1.0) GO TO 185
      IF(S(J,2).EQ.S(J3,3)) GO TO 181
      IF(S(J,3).EQ.S(J3,3)) GO TO 182
      IF(S(J,2).EQ.S(J4,3)) GO TO 183
      IF(S(J,2).EQ.S(I,3)) GO TO 184
      GO TO 185
  181 J8=J
      K8=K8+1
      IF(K8)185,185,190
  182 IF(S(J,2).EQ.S(J2,2)) GO TO 190
      J9=J
      K9=K9+1
      IF(K9)185,185,190
  183 J12=J
      K12=K12+1
      GO TO 185
  184 IF(S(J,3).EQ.S(J1,3)) GO TO 190
      J13=J
      K13=K13+1
      IF(K13)185,185,190
  185 CONTINUE
      IF(K13.EQ.0.AND.K12.EQ.0.AND.K8.EQ.-1.AND.K9.EQ.-1) GO TO 186
      IF(K13.EQ.-1.AND.K12.EQ.0.AND.K8.EQ.0.AND.K9.EQ.-1) GO TO 187
      IF(K13.EQ.-1.AND.K9.EQ.0.AND.K8.EQ.-1) GO TO 188
      IF(K13.EQ.-1.AND.K9.EQ.-1.AND.K8.EQ.0) GO TO 189
      IF(K13.EQ.-1.AND.K9.EQ.-1.AND.K8.EQ.-1) GO TO 900
      GO TO 190
C
C     CRISS CROSS 1
C
  186 IF(S(J13,3).NE.S(J12,3)) GO TO 190
      DO 192 J=1,NK
      IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J2.OR.J.EQ.J3.OR.J.EQ.J4.OR.J.EQ.J12.
     1OR.J.EQ.J13.OR.S(J,4).LE.0) GO TO 192
      IF(S(J,3).EQ.S(J4,3)) GO TO 190
  192 CONTINUE
       GO TO 400
C
C     CRISS CROSS 2
C
  187 IF(S(J8,3).NE.S(J12,3)) GO TO 190
       DO 193 J=1,NK
       IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J2.OR.J.EQ.J3.OR.J.EQ.J4.OR.J.EQ.J8.
     1OR.J.EQ.J9.OR.S(J,4).LE.0) GO TO 193
      IF(S(J,3).EQ.S(J4,3)) GO TO 190
  193 CONTINUE
      GO TO 500
C
C     DOUBLE WHEATSTONE BRIDGE 1
```

```
C
  188 K10=-1
      K11=-1
      DO 196 J=1,NK
      IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J2.OR.J.EQ.J3.OR.J.EQ.J4.OR.J.EQ.J9)
     1GO TO 196
      IF(S(J,3).EQ.S(J9,2)) GO TO 194
      IF(S(J,2).EQ.S(J9,2)) GO TO 195
      GO TO 196
  194 IF(S(J,2).NE.S(J3,2)) GO TO 190
      J10=J
      K10=K10+1
      IF(K10)196,196,190
  195 IF(S(J,3).NE.S(J4,3)) GO TO 190
      J11=J
      K11=K11+1
      IF(K11)196,196,190
  196 CONTINUE
      IF(K10.EQ.0.AND.K11.EQ.0) GO TO 700
      GO TO 190
C
C     DOUBLE WHEATSTONE BRIDGE 2
C
  189 K10=-1
      K11=-1
      DO 199 J=1,NK
      IF(J.EQ.I.OR.J.EQ.J1.OR.J.EQ.J2.OR.J.EQ.J3.OR.J.EQ.J4.OR.J.EQ.J8 .
     1OR.S(J,4).LE.0) GO TO 199
      IF(S(J,3).EQ.S(J8,3)) GO TO 197
      IF(S(J,2).EQ.S(J8,3)) GO TO 198
      GO TO 199
  197 IF(S(J,2).NE.S(J3,2)) GO TO 190
      J10=J
      K10=K10+1
      IF(K10)199,199,190
  198 IF(S(J,3).NE.S(J4,3)) GO TO 190
      J11=J
      K11=K11+1
      IF(K11)199,199,190
  199 CONTINUE
      IF(K10.EQ.0.AND.K11.EQ.0) GO TO 800
  190 CONTINUE
      IF(NRED)1103,1103,91
   91 NRED = 0
      GO TO 100
C
C  BEGINNING OF PARALLEL REDUCTION SECTION
C
  200 NRED=NRED+1
      J1=JP(1)
      DO 201 J=2,KP
      L=JP(J)
201   S(L,4)=-1
      S(I,4)=-1
      ISBR=1
      LIST(23)=KP
      LIST(24)=J1
      WRITE(F7) LIST
      IF (JMAT) 1040,1040,120
 1040 LIM=KP-1
      DO 202 K=1,KP
```

```
      IF=JP(K)
  202 JP(K)=S(IP,1)
      WRITE(6,1024) S(I,1),(JP(K),K=1,LIM)
      WRITE(6,1023) JP(KP)
 1024 FORMAT(1H0, 10X11H ACTIVITIES I4,(10(2H, I4)/12X))
 1023 FORMAT(1H ,10X4HAND I4,32H HAVE BEEN COMBINED IN PARALLEL.    )
      GO TO 1290
C
C     SERIES REDUCTION
C
  300 NRED=NRED+1
      S(I,4)=-1
      ISER=2
      LIST(3)=J1
      WRITE(F7) LIST
      IF (JMAT) 1009,1009,120
 1009 WRITE(6,1025) S(I,1),S(J1,1)
 1025 FORMAT(1H0,42X11H ACTIVITIES I4,4H AND I4,20H COMBINED IN SERIES.)
      GO TO 1290
C
C     CRISS CROSS TYPE 1 REDUCTION
C
  400 NRED=NRED+1
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J12,4)=-1
      S(J13,4)=-1
      ISBR=3
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J4
      LIST(7)=J12
      LIST(8)=J13
      WRITE(F7) LIST
      IF(JMAT) 1230,1230,190
 1230 WRITE(6,1430)S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J4,1),S(J12,1),S(J13
     1,1)
 1430 FORMAT(1H0,25X11HACTIVITIES I4,2H, I4,2H, I4,2H, I4,2H, I4,2H, I4,
     1  6H, AND I4,44H HAVE BEEN COMBINED AS A CRISS CROSS TYPE 1.)
      GO TO 1290
C
C     CRISS CROSS TYPE 2 REDUCTION
C
  500 NRED=NRED+1
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J8,4)=-1
      S(J12,4)=-1
      ISBR=4
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J4
      LIST(7)=J8
      LIST(8)=J12
      WRITE(F7) LIST
```

```
      IF(JMAT) 1240,1240,190
 1240 WRITE(6,1440)S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J4,1),S(J8,1),S(J12,
     11)
 1440 FORMAT(1H0,25X 11HACTIVITIES I4,2H, I4,2H, I4,2H, I4,2H, I4,2H, I4
     1, 6H, AND I4,44H HAVE BEEN COMBINED AS A CRISS CROSS TYPE 2.)
      GO TO 1290
C
C     CRISS CROSS TYPE 3 REDUCTION
C
  600 NRED=NRED+1
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J5,4)=-1
      S(J6,4)=-1
      S(J7,4)=-1
      ISBR=5
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J5
      LIST(7)=J6
      LIST(8)=J7
      WRITE(F7) LIST
      IF(JMAT) 1250,1250,190
 1250 WRITE(6,1450)S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J5,1),S(J6,1),S(J7,1
     1)
 1450 FORMAT(1H0,25X 11HACTIVITIES I4,2H, I4,2H, I4,2H, I4,2H, I4,2H, I4
     1, 6H, AND I4,44H HAVE BEEN COMBINED AS A CRISS CROSS TYPE 3.)
      GO TO 1290
C
C     DOUBLE WHEATSTONE BRIDGE TYPE 1 REDUCTION
C
  700 NRED=NRED+1
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J9,4)=-1
      S(J10,4)=-1
      S(J11,4)=-1
      ISBR=6
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J4
      LIST(7)=J9
      LIST(8)=J10
      LIST(9)=J11
      WRITE(F7) LIST
      IF(JMAT) 1210,1210,190
 1210 WRITE(6,1410)S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J4,1),S(J9,1),S(J10,
     11),S(J11,1)
 1410 FORMAT(1H0,10X 11HACTIVITIES I4,2H, I4,2H, I4,2H, I4,2H, I4,2H, I4
     1,2H, I4,6H, AND I4, 57H HAVE BEEN COMBINED AS A DOUBLE WHEATSTONE
     2BRIDGE TYPE 1.)
      GO TO 1290
C
C      DOUBLE WHEATSTONE BRIDGE TYPE 2 REDUCTION
C
  800 NRED=NRED+1
```

```
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J8,4)=-1
      S(J10,4)=-1
      S(J11,4)=-1
      ISER=7
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J4
      LIST(7)=J8
      LIST(8)=J10
      LIST(9)=J11
      WRITE(F7) LIST
      IF(JMAT) 1220,1220,190
 1220 WRITE(6,1420)S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J4,1),S(J8,1),S(J10,
     11),S(J11,1)
 1420 FORMAT(1H0,10X 11HACTIVITIES I4,2H, I4,2H, I4,2H, I4,2H, I4,2H, I4
     1,2H, I4,6H, AND I4, 57H HAVE BEEN COMBINED AS A DOUBLE WHEATSTONE
     2BRIDGE TYPE 2.)
      GO TO 1290
C
C     WHEATSTONE BRIDGE REDUCTION
C
  900 IF (NRED.EQ.0) NRED=NRED+1
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      ISER=8
      LIST(3)=J1
      LIST(4)=J2
      LIST(5)=J3
      LIST(6)=J4
      WRITE(F7) LIST
      IF(JMAT) 1201,1201,190
 1201 WRITE(6,1401) S(I,1),S(J1,1),S(J2,1),S(J3,1),S(J4,1)
 1401 FORMAT(1H0,20X 11HACITIVITES I4,2H, I4,2H, I4,2H, I4,6H, AND I4, 4
     13H HAVE BEEN COMBINED AS A WHEATSTONE BRIDGE.)
C
 1290 WRITE(6,1490) S(J1,1),S(J1,2),S(J1,3)
 1490 FORMAT(1H ,41X, 9HACTIVITY ,I4,18H NOW HAS TAIL NO. I4,14H AND HEA
     1D NO. ,I4,1H.)
      GO TO 190
 1103 WRITE(6,1027)
 1027 FORMAT('1',10X,'THE FOLLOWING ACTIVITIES CANNOT BE FURTHER COMBINE
     *D:')
      WRITE(6,2001)
 2001 FORMAT(///11X,'ACTIVITY     ORIGIN     TERMINAL'
     *5X,'ACTIVITY NUMBER ASSIGNED BY DECOMP')
      NACT=0
      DO 1105 I=1,NK
      IF(S(I,4).LT.0)GO TO 1105
      NACT=NACT+1
 1104 WRITE(6,1028)S(I,1),S(I,2),S(I,3),NACT
 1028 FORMAT(13X,I4,6X,I5,7X,I5,9X,I5)
 1105 CONTINUE
      ENDFILE F7
C
```

```
C          BUILD FILE3 FOR DECOMPOSITION
C
      NACT = 0
      LNODEN = 0
C
C     COUNT ACTIVITIES AND DETERMINE LARGEST NODE NUMBER
C     DETERMINE L U AND P FOR SUBNETWORK ANALYSIS
C
      DO 1153 J=1,NK
C
C     S(J,4) IS -1 IF J-TH ACTIVITY HAS BEEN ELIMINATED
C
      IF(S(J,4).EQ.-1)GO TO 1153
      NACT=NACT+1
      S(NACT,1)=NACT
      S(NACT,2)=S(J,2)
      S(NACT,3)=S(J,3)
      IF (S(NACT,3).GT.LNODEN) LNODEN = S(NACT,3)
1153  CONTINUE
C
C     FIND SOURCE AND SINK
C
      NSRCE = 0
      NSINK = 0
      DO 1154  J=1,NACT
      IFLG = 0
      JFLG  = 0
      IIRG = S(J,2)
      ITER = S(J,3)
      DO 1156  K=1,NACT
      IF (IIRG.EQ.S(K,3)) IFLG=1
      IF (ITER.EQ.S(K,2)) JFLG=1
1156  CONTINUE
      IF (IFLG.EQ.0) NSRCE = S(J,2)
      IF (JFLG.EQ.0) NSINK = S(J,3)
      IF (NSRCE.NE.0.AND.NSINK.NE.0) GO TO 1157
1154  CONTINUE
1157  CONTINUE
      WRITE(F3)NACT,NODES,NSRCE,NSINK,LNODEN,FILE3
      STOP
      END
```

5. MODSIMP

```
C       MODIFIED SIMPLIFICATION PROGRAM
C
C           THIS PROGRAM DETERMINES THE DISTRIBUTION OF EACH OF THE
C           ACTIVITIES IN THE SIMPLIFIED PROJECT NETWORK (DETERMINED BY THE
C           SIMPLIFICATION PROGRAM).
C
C           FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C           MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                       PROJECT NETWORK
C           CURRENTLY, MMAX=1000
C
C       DIMENSION S(MMAX,4),CDF(MMAX,12)
C
        IMPLICIT REAL*8(A-H,O-R,T-Z)
        INTEGER F0/8/,      F2/10/,         F4/12/,         F7/15/
        COMMON /BLKA/S,CDF,A,DENS,TI
        INTEGER S
        DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
        COMMON /BLKB/TIME,Y
        DIMENSION TIME(4),Y(12)
        REAL*4 TYME(4)
        DIMENSION             JP(20),LIST(24)
        EQUIVALENCE (ISBR,LIST(1)),(I,LIST(2)),(JP(1),LIST(3))
        WRITE(6,9000)
9000    FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE MODI
       *FIED SIMPLIFICATION PROGRAM: MODSIMP'/1H0,132('*')/1X,132('*'))
        REWIND F0
        REWIND F2
        REWIND F7
        A(1)=0.0
        A(12)=1.00
        DO 82 J=2,11
     82 A(J)= DFLOAT(J-2)*0.1D0 + 0.05D0
        I500=0
        J200=0
        READ(F0)NACT,NODES,NSRCE,NSINK,LNODEN,S
        IFLOP = 0
      5 DO 79 I=1,NACT
        READ(F2)MM,TYME
        MM=MM+2
        TIME(1) = TYME(1)
        TIME(2) = TYME(2)
        TIME(3) = TYME(3)
        TIME(4) = TYME(4)
        CALL XDENS(MM,I,I500,J200)
        S(I,4)=MM
        DO 42 J=1,12
42      CDF(I,J)=Y(J)
     79 CONTINUE
        NK=NACT
100     READ(F7,END=99) LIST
        GO TO (200,300,400,500,600,700,800,900),ISBR
C
200     CONTINUE
C       PARALLEL REDUCTION
        KP = LIST(23)
        J1 = LIST(24)
        CALL PARRED(I,JP,KP,J1)
        GO TO 100
300     CONTINUE
```

```
C
C     SERIES REDUCTION
C
      J1 = LIST(3)
      S(I,4)=-1
      CALL SERRED(I,J1)
      GO TO 100
400   CONTINUE
C
C     CRISS CROSS TYPE 1 REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J4 = LIST(6)
      J12= LIST(7)
      J13= LIST(8)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J12,4)=-1
      S(J13,4)=-1
      CALL CC1RED(I,J1,J2,J3,J4,J12,J13)
      GO TO 100
500   CONTINUE
C
C     CRISS CROSS TYPE 2 REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J4 = LIST(6)
      J8 = LIST(7)
      J12= LIST(8)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J8,4)=-1
      S(J12,4)=-1
      CALL CC2RED(I,J1,J2,J3,J4,J8,J12)
      GO TO 100
600   CONTINUE
C
C     CRISS CROSS TYPE 3 REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J5 = LIST(6)
      J6 = LIST(7)
      J7 = LIST(8)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J5,4)=-1
      S(J6,4)=-1
      S(J7,4)=-1
      CALL CC3RED(I,J1,J2,J3,J5,J6,J7)
      GO TO 100
```

```
700   CONTINUE
C
C     DOUBLE WHEATSTONE BRIDGE TYPE 1 REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J4 = LIST(6)
      J9 = LIST(7)
      J10= LIST(8)
      J11= LIST(9)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J9,4)=-1
      S(J10,4)=-1
      S(J11,4)=-1
      CALL DW1RED(I,J1,J2,J3,J4,J9,J10,J11)
      GO TO 100
800   CONTINUE
C
C      DOUBLE WHEATSTONE BRIDGE TYPE 2 REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J4 = LIST(6)
      J8 = LIST(7)
      J10= LIST(8)
      J11= LIST(9)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      S(J8,4)=-1
      S(J10,4)=-1
      S(J11,4)=-1
      CALL DW2RED(I,J1,J2,J3,J4,J8,J10,J11)
      GO TO 100
900   CONTINUE
C
C     WHEATSTONE BRIDGE REDUCTION
C
      J1 = LIST(3)
      J2 = LIST(4)
      J3 = LIST(5)
      J4 = LIST(6)
      S(I,4)=-1
      S(J2,4)=-1
      S(J3,4)=-1
      S(J4,4)=-1
      CALL WBRED(I,J1,J2,J3,J4)
99    CONTINUE
C     PRINT RESULTING DISTRIBUTIONS
      DO 1100 I=1,NACT
      IF (S(I,4).LT.0) GO TO 1100
      TIME(1)=0.D0
      TIME(3)=0.D0
      DO 270 K=2,11
      TIME(1)=TIME(1)+CDF(I,K)
```

```
270   TIME(3)=TIME(3)+CDF(I,K)**2
      TIME(1)=TIME(1)*.1D0
      TIME(3)=DSQRT(DMAX1(0.D0,TIME(3)*.1D0-TIME(1)**2))
      IFLOP = IFLOP + 1
      IF (MOD(IFLOP,3).EQ.0) WRITE(6,2001)
2001  FORMAT(1H1)
      WRITE(6,1028)S(I,1),S(I,2),S(I,3),TIME(1),TIME(3)
1028  FORMAT(///11X,'ACTIVITY     ORIGIN     TERMINAL      MEAN      STAND
     *ARD DEVIATION'/13X,I4,6X,I5,7X,I5,6X,F6.1, 8X,F7.2//32X,'F(T)
     * T')
      DO 1102 MP=1,12
      WRITE(6,6000) A(MP),CDF(I,MP)
6000  FORMAT(31XF5.2,3H --   F10.4)
1102  CONTINUE
1100  CONTINUE
C
C     SAVE THE SIMPLIFIED ACTIVITY DURATION DISTRIBUTIONS FOR SUBNETWORK
C     ANALYSIS
C
      REWIND F4
      READ(F4)
      NACT=0
      DO 11002 J=1,NK
      IF(S(J,4).GE.0)NACT=NACT+1
11002 CONTINUE
      WRITE(F4) NACT
      DO 1153 J=1,NK
      IF(S(J,4).GT.0)WRITE(F4)(CDF(J,K),K=1,12)
1153  CONTINUE
      ENDFILE F4
C     END OF MAIN PROGRAM
C
      STOP
      END
      SUBROUTINE PARRED(I,JP,KP,J1)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      DIMENSION JP(20)
      TI(1)=CDF(I,1)
      TI(2)=CDF(I,12)
      J1=JP(1)
      DO 201 J=1,KP
      L=JP(J)
      IF(CDF(L,1).GT.TI(1)) TI(1)=CDF(L,1)
      IF(CDF(L,12).GT.TI(2)) TI(2)=CDF(L,12)
      IF(L.EQ.J1) GO TO 201
      S(L,4)=-1
  201 CONTINUE
      S(I,4)=-1
  213 L=I
      KM=10
      K=1
      DELTAT=(TI(2)-TI(1))/10.0D0
      MP=0
      M=1
  217 DO 220 J=2,10
      K=K+1
      T=TI(1)+DFLOAT(J-1)*DELTAT
      IF(T.GE.CDF(L,12)) GO TO 221
```

```
  218 IF(T.GE.CDF(L,M).AND.T.LT.CDF(L,M+1)) GO TO 219
      M=M+1
      GO TO 218
  219 IF(M.NE.1) GO TO260
      T1=CDF(L ,1)
      T2=CDF(L ,2)
      T3=CDF(L ,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 261
  260 T1=CDF(L,M)
      T2=CDF(L,M+1)
      T3=CDF(L,M-1)
      Y1=A(M)
      Y2=A(M+1)
      Y3=A(M-1)
  261 DENS(K)=TERPOL(Y1,Y2,Y3,T,T1,T2,T3)
      IF(DENS(K).LT.0.D0) DENS(K)=0.D0
      IF(DENS(K).GE.1.D0) GO TO 221
  220 CONTINUE
      GO TO 231
  221 KO=K
      DO 230 K=KO,KM
  230 DENS(K)=1.D0
  231 MP=MP+1
      IF(MP.GT.KP) GO TO 240
      L=JP(MP)
      K=KM
      M=1
      KM=KM+9
      GO TO 217
 240  CONTINUE
      DO 251 M=2,10
      DO 250 J=1,KP
      L=M+J*9
  250 DENS(M)=DENS(M)*DENS(L)
  251 CONTINUE
      DENS(1)=0.0D0
      DENS(11)=1.0D0
      CALL INVSTR(J1,DELTAT)
      RETURN
      END
      SUBROUTINE SERRED(I,J1)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      TI(1)=CDF(I,1)+CDF(J1,1)
      TI(2)=CDF(I,12)+CDF(J1,12)
      IF(CDF(J1,12).EQ.CDF(J1,1)) GO TO 370
      IF(CDF(I,12).EQ.CDF(I ,1)) GO TO 376
      DENS(1)=0.D0
      DENS(11)=1.0D0
      TEMPO=TI(1)
      DELTAT=(TI(2)-TI(1))/10.0D0
      DO 399 J=2,10
      K1=1
      T=TEMPO+DFLOAT(J-1)*DELTAT
      FOT=0.0D0
      LI=11
```

```
  311 K1=K1+1
      IF(K1.GE.12) GO TO 390
      TDIF=T-CDF(I .K1)
      IF(TDIF.GE.CDF(J1.12)) GO TO 350
  312 IF (TDIF.GT.CDF(J1.LI)) GO TO 340
      LI=LI-1
      IF(1-LI)312.312.390
  340 IF (LI .NE. 1) GO TO 360
      T1=CDF(J1.1)
      T2=CDF(J1.2)
      T3=CDF(J1.3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GC TC 361
  360 T1=CDF(J1.LI)
      T2=CDF(J1.LI+1)
      T3=CDF(J1.LI-1)
      Y1=A(LI)
      Y2=A(LI+1)
      Y3=A(LI-1)
  361 FOT=FOT+TERPOL(Y1.Y2.Y3.TDIF.T1.T2.T3)
      GO TO 311
  350 FOT=FOT+1.D0
      GC TC 311
  390 DENS(J)=0.1D0*FOT
      IF(DENS(J).GE.1.D0) GO TO 380
      IF(DENS(J).GT.0.D0) GO TO 399
      DENS(J)=0.D0
  399 CONTINUE
      GO TO 382
  380 DO 381 JJ=J.10
      DENS(JJ)=1.D0
  381 CONTINUE
  382 S(J1.2)=S(I.2)
  369 CONTINUE
      CALL INVSTR(J1.DELTAT)
      GO TO 379
  370 DO  372 JJ=1.12
  372 CDF(J1.JJ)= CDF(I.JJ) + CDF(J1.12)
      S(J1.2)=S(I.2)
      GO TO 379
  376 DO 378 JJ=1.12
  378 CDF(J1.JJ)=CDF(J1.JJ) + CDF(I.12)
      S(J1.2)= S(I.2)
 379  RETURN
      END
      SUBROUTINE CC1RED(I.J1.J2.J3.J4.J12.J13)
      IMPLICIT REAL*8(A-H.O-R.T-Z)
      COMMON /BLKA/S.CDF.A.DENS.TI
      INTEGER S
      DIMENSION S(1000.4).CDF(1000.12).A(12).DENS(400).TI(2)
      DO 406 J=1.2
      L=1+(J-1)*11
      T1=CDF(J3.L)+CDF(J4.L)+CDF(J12.L)
      T2=CDF(I.L)+CDF(J2.L)+CDF(J12.L)+CDF(J4.L)
      T3=CDF(I.L)+CDF(J1.L)+CDF(J12.L)
      T4=CDF(I.L)+CDF(J13.L)
      IF(T1.GE.T2.AND.T1.GE.T3.AND.T1.GE.T4) GO TO 403
      IF(T2.GE.T1.AND.T2.GE.T3.AND.T2.GE.T4) GC TO 404
      IF(T3.GE.T1.AND.T3.GE.T2.AND.T3.GE.T4) GO TO 405
```

```
      TI(J)=T4
      GO TO 406
  403 TI(J)=T1
      GO TO 406
  404 TI(J)=T2
      GO TO 406
  405 TI(J)=T3
  406 CONTINUE
      DELTAT=(TI(2)-TI(1))/10.0D0
      DENS(1)=0.D0
      DENS(11)=1.0D0
      DO 470 J=2,10
      T=TI(1) + DFLOAT(J-1)*DELTAT
      KI=1
      L13=11
      FCT=0.0
  410 KI=KI+1
      IF(KI.GE.12) GO TO 460
      TDIF13=T-CDF(I,KI)
      IF(TDIF13.GE.CDF(J13,12)) GO TO 412
  411 IF(TDIF13.GT.CDF(J13,L13)) GO TO 413
      L13=L13-1
      IF(1-L13)411,411,460
  412 F13=1.D0
      GO TO 415
  413 IF (L13.NE. 1) GO TO 491
      T1=CDF(J13,1)
      T2=CDF(J13,2)
      T3=CDF(J13,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 414
  491 T1=CDF(J13,L13)
      T2=CDF(J13,L13+1)
      T3=CDF(J13,L13-1)
      Y1=A(L13)
      Y2=A(L13+1)
      Y3=A(L13-1)
  414 F13=TERPOL(Y1,Y2,Y3,TDIF13,T1,T2,T3)
      IF(F13.LT.0.D0)F13=0.D0
      IF(F13.GT.1.D0)F13=1.D0
  415 K12=1
      L1=11
      FCT2=0.D0
  416 K12=K12+1
      IF(K12.LT.12) GO TO 418
  417 FCT=FCT+FCT2*F13
      GO TO 410
  418 TDIF1=TDIF13-CDF(J12,K12)
      IF(TDIF1.GE.CDF(J1,12)) GO TO 420
  419 IF(TDIF1.GT.CDF(J1,L1)) GO TO 421
      L1=L1-1
      IF(1-L1)419,419,417
  420 F1=1.D0
      GO TO 422
  421 IF (L1 .NE. 1) GO TO 482
      T1=CDF(J1,1)
      T2=CDF(J1,2)
      T3=CDF(J1,3)
      Y1=0.D0
```

```
      Y2=0.05D0
      Y3=0.15D0
      GO TO 427
  482 T1=CDF(J1,L1)
      T2=CDF(J1,L1+1)
      T3=CDF(J1,L1-1)
      Y1=A(L1)
      Y2=A(L1+1)
      Y3=A(L1-1)
  427 F1=TERPOL(Y1, Y2, Y3, TDIF1, T1, T2, T3)
      IF(F1.LT.0.D0) F1=0.D0
      IF(F1.GT.1.D0) F1=1.D0
  422 K4=1
      L3=11
      L2=11
      FOT1=0.D0
  423 K4=K4+1
      IF(K4.LT.12) GO TO 425
  424 FOT2=FOT2+F1*FOT1
      GO TO 416
  425 TDIF3=T-CDF(J4,K4)-CDF(J12,K12)
      TDIF2=TDIF3-CDF(I,KI)
      IF(TDIF2.GE.CDF(J2,12)) GO TO 430
  426 IF(TDIF2.GT.CDF(J2,L2)) GO TO 431
      L2=L2-1
      IF(1-L2)426,426,424
  430 F2=1.D0
      GO TO 435
  431 IF (L2 .NE. 1) GO TO 433
      T1=CDF(J2,1)
      T2=CDF(J2,2)
      T3=CDF(J2,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 434
  433 T1=CDF(J2,L2)
      T2=CDF(J2,L2+1)
      T3=CDF(J2,L2-1)
      Y1=A(L2)
      Y2=A(L2+1)
      Y3=A(L2-1)
  434 F2=TERPOL(Y1, Y2, Y3, TDIF2, T1, T2, T3)
      IF(F2 .LT.0.D0) F2=0.D0
      IF(F2 .GT.1.D0) F2=1.D0
  435 IF(TDIF3.GE.CDF(J3,12)) GO TO 440
  436 IF(TDIF3.GT.CDF(J3,L3)) GO TO 445
      L3=L3-1
      IF(1-L3)436,436,424
  440 F3=1.D0
      GO TO 450
  445 IF (L3 .NE. 1) GO TO 447
      T1=CDF(J3,1)
      T2=CDF(J3,2)
      T3=CDF(J3,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 449
  447 T1=CDF(J3,L3)
      T2=CDF(J3,L3+1)
```

```
      T3=CDF(J3,L3-1)
      Y1=A(L3)
      Y2=A(L3+1)
      Y3=A(L3-1)
  449 F3=TERPOL(Y1, Y2, Y3, TDIF3, T1, T2, T3)
      IF(F3 .LT.0.D0) F3=0.D0
      IF(F3.GT.1.D0) F3=1.D0
  450 FCT1=FCT1+F2*F3
      GC TC 423
  460 DENS(J)=FCT*0.1D-2
      IF(DENS(J).LT.0.D0) DENS(J)=0.D0
      IF(DENS(J).GE.1.D0) GO TO 480
  470 CONTINUE
  480 DC 481 JJ=J,10
      DENS(JJ)=1.D0
  481 CCNTINUE
      S(J1,3)=S(J13,3)
      S(J1,2)=S(I,2)
      CALL INVSTR(J1,DELTAT)
      RETURN
      ENC
      SUBROUTINE CC2RED(I,J1,J2,J3,J4,J8,J12)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      CCMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      DO 510 J=1,2
      L=1+(J-1)*11
      T1=CDF(I,L)+CDF(J1,L)+CDF(J12,L)
      T2=CDF(I,L)+CDF(J2,L)+CDF(J4,L)+CDF(J12,L)
      T3=CDF(I,L)+CDF(J2,L)+CDF(J8,L)
      T4=CDF(J3,L)+CDF(J4,L)+CDF(J12,L)
      T5=CDF(J3,L)+CDF(J8,L)
      IF(T1.GE.T2.AND.T1.GE.T3.AND.T1.GE.T4.ANC.T1.GE.T5) GO TO 501
      IF(T2.GE.T1.AND.T2.GE.T3.AND.T2.GE.T4.AND.T2.GE.T5) GO TO 502
      IF(T3.GE.T1.AND.T3.GE.T2.AND.T3.GE.T4.AND.T3,GE.T5) GO TO 503
      IF(T4.GE.T1.AND.T4.GE.T2.AND.T4.GE.T3.AND.T4.GE.T5) GO TO 504
      TI(J)=T5
      GO TO 510
  501 TI(J)=T1
      GC TO 510
  502 TI(J)=T2
      GO TO 510
  503 TI(J)=T3
      GO TO 510
  504 TI(J)=T4
  510 CCNTINUE
      DELTAT=(TI(2)-TI(1))/10.0D0
      DENS(1)=0.D0
      DENS(11)=1.0D0
      DO 570 J=2,10
      T=TI(1) + DFLOAT(J-1)*DELTAT
      KI=1
      FOT=0.D0
  511 KI=KI+1
      IF(KI.GE.12) GO TO 560
      K12=1
      L1=11
      FOT3=0.D0
  512 K12=K12+1
      IF(K12.LT.12) GO TO 514
```

```
513 FCT=FCT+FCT3
    GO TO 511
514 TDIF1=T-CDF(I,KI)-CDF(J12,K12)
    IF(TDIF1.GE.CDF(J1,12)) GO TO 517
515 IF(TDIF1.GT.CDF(J1,L1)) GO TO 518
    L1=L1-1
    IF(1-L1)515,515,513
517 F1=1.D0
    GO TO 519
518 IF (L1 .NE. 1) GO TO 971
    T1=CDF(J1,1)
    T2=CDF(J1,2)
    T3=CDF(J1,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 972
971 T1=CDF(J1,L1)
    T2=CDF(J1,L1+1)
    T3=CDF(J1,L1-1)
    Y1=A(L1)
    Y2=A(L1+1)
    Y3=A(L1-1)
972 F1=TERPOL(Y1, Y2, Y3, TDIF1, T1, T2, T3)
    IF(F1.LT.0.D0) F1=0.D0
    IF(F1.GT.1.D0) F1=1.D0
519 K3=1
    FCT2=0.D0
520 K3=K3+1
    IF(K3.LT.12) GO TO 521
    FCT3=FCT3+F1*FCT2
    GO TO 512
521 K2=1
    FCT1=0.D0
    L4=11
    L8=11
522 K2=K2+1
    IF(K2.LT.12) GO TO 526
525 FCT2=FCT2+FCT1
    GO TO 520
526 T1=T-CDF(J3,K3)
    T2=T-CDF(I,KI)-CDF(J2,K2)
    IF(T2-T1)528,527,527
528 TDIF8=T2
    GO TO 530
527 TDIF8=T1
    TDIF4=TDIF8-CDF(J12,K12)
530 IF(TDIF4.GE.CDF(J4,12)) GO TO 535
531 IF(TDIF4.GT.CDF(J4,L4)) GO TO 540
    L4=L4-1
535 F4=1.D0
    GO TO 541
540 IF (L4 .NE. 1) GO TO 973
    T1=CDF(J4,1)
    T2=CDF(J4,2)
    T3=CDF(J4,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 974
973 T1=CDF(J4,L4)
```

```
      T2=CDF(J4,L4+1)
      T3=CDF(J4,L4-1)
      Y1=A(L4)
      Y2=A(L4+1)
      Y3=A(L4-1)
  974 F4=TERPOL(Y1, Y2, Y3, TDIF4, T1, T2, T3)
      IF(F4 .LT.0.D0) F4=0.D0
      IF(F4 .GT.1.D0) F4=1.D0
  541 IF(TDIF8.GE.CDF(J8,12)) GO TO 545
  542 IF(TDIF8.GT.CDF(J8,L8)) GO TO 550
      L8=L8-1
      IF(1-L8) 542,542,525
  545 F8=1.D0
      GO TO 551
  550 IF (L8 .NE. 1) GO TO 975
      T1=CDF(J8,1)
      T2=CDF(J8,2)
      T3=CDF(J8,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 976
  975 T1=CDF(J8,L8)
      T2=CDF(J8,L8+1)
      T3=CDF(J8,L8-1)
      Y1=A(L8)
      Y2=A(L8+1)
      Y3=A(L8-1)
  976 F8=TERPOL(Y1, Y2, Y3, TDIF8, T1, T2, T3)
      IF(F8 .LT.0.D0) F8=0.D0
      IF(F8. GT.1.D0) F8=1.D0
  551 FOT1=FOT1+F4*F8
      GO TO 522
  560 DENS(J)=FOT*0.1D-3
      IF(DENS(J).LT.0.D0) DENS(J)=0.D0
      IF(DENS(J).GE.1.D0) GO TO 580
  570 CONTINUE
  580 DO 581 JJ=J,10
      DENS(JJ)=1.D0
  581 CONTINUE
      S(J1,2)=S(I,2)
      S(J1,3)=S(J12,3)
      CALL INVSTR(J1,DELTAT)
      RETURN
      END
      SUBROUTINE CC3RED(I,J1,J2,J3,J5,J6,J7)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      DO 610 J=1,2
      L=1+(J-1)*11
      T1=CDF(I,L)+CDF(J1,L)+CDF(J6,L)
      T2=CDF(I,L)+CDF(J2,L)+CDF(J7,L)
      T3=CDF(I,L)+CDF(J1,L)+CDF(J5,L)+CDF(J7,L)
      T4=CDF(J3,L)+CDF(J7,L)
      IF(T1.GE.T2.AND.T1.GE.T3.AND.T1.GE.T4) GO TO 602
      IF(T2.GE.T1.AND.T2.GE.T3.AND.T2.GE.T4) GO TO 603
      IF(T3.GE.T1.AND.T3.GE.T2.AND.T3.GE.T4) GO TO 604
      TI(J)=T4
      GO TO 610
```

```
602 TI(J)=T1
    GO TO 610
603 TI(J)=T2
    GO TO 610
604 TI(J)=T3
    GO TO 610
610 CONTINUE
    DELTAT=(TI(2)-TI(1))/10.0D0
    DENS(1)=0.D0
    DENS(11)=1.0D0
    DO 670 J=2,10
    T=TI(1) + DFLOAT(J-1)*DELTAT
    K7=1
    L3=11
    FOT=0.D0
615 K7=K7+1
    IF(K7.GE.12) GO TO 660
    TDIF3=T-CDF(J7,K7)
    IF(TDIF3.GE.CDF(J3,12)) GO TO 625
620 IF(TDIF3.GT.CDF(J3,L3)) GO TO 622
    L3=L3-1
    IF(1-L3)620,620,660
622 IF (L3 .NE. 1) GO TO 671
    T1=CDF(J3,1)
    T2=CDF(J3,2)
    T3=CDF(J3,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 672
671 T1=CDF(J3,L3)
    T2=CDF(J3,L3+1)
    T3=CDF(J3,L3-1)
    Y1=A(L3)
    Y2=A(L3+1)
    Y3=A(L3-1)
672 F3=TERPOL(Y1, Y2, Y3, TDIF3, T1, T2, T3)
    IF(F3 .LT.0.D0) F3=0.D0
    IF(F3.GT.1.D0) F3=1.D0
    GO TO 626
625 F3=1.D0
626 KI=1
    L2=11
    FOT2=0.D0
627 KI=KI+1
    IF(KI.LT.12) GO TO 630
628 FOT=FOT+F3*FOT2
    GO TO 615
630 TDIF2=TDIF3-CDF(I,KI)
    IF(TDIF2.GE.CDF(J2,12)) GO TO 635
631 IF(TDIF2.GT.CDF(J2,L2)) GO TO 636
    L2=L2-1
    IF(1-L2)631,631,628
635 F2=1.D0
    GO TO 637
636 IF (L2 .NE. 1) GO TO 673
    T1=CDF(J2,1)
    T2=CDF(J2,2)
    T3=CDF(J2,3)
    Y1=0.D0
    Y2=0.05D0
```

```
      Y3=0.15D0
      GO TO 674
  673 T1=CDF(J2,L2)
      T2=CDF(J2,L2+1)
      T3=CDF(J2,L2-1)
      Y1=A(L2)
      Y2=A(L2+1)
      Y3=A(L2-1)
  674 F2=TERPOL(Y1, Y2, Y3, TDIF2, T1, T2, T3)
      IF(F2 .LT.0.D0) F2=0.D0
      IF(F2 .GT.1.D0) F2=1.D0
  637 K1=1
      FCT1=0.D0
      L5=11
      L6=11
  640 K1=K1+1
      IF(K1.LT.12) GO TO 645
  641 FCT2=FCT2+F2*FCT1
      GO TO 627
  645 TDIF5=TDIF2-CDF(J1,K1)
      TDIF6=T-CDF(I,K1)-CDF(J1,K1)
      IF(TDIF5.GE.CDF(J5,12)) GO TO 650
  646 IF(TDIF5.GT.CDF(J5,L5)) GO TO 647
      L5=L5-1
      IF(1-L5)646,646,641
  647 IF (L5 .NE. 1) GO TO 675
      T1=CDF(J5,1)
      T2=CDF(J5,2)
      T3=CDF(J5,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 676
  675 T1=CDF(J5,L5)
      T2=CDF(J5,L5+1)
      T3=CDF(J5,L5-1)
      Y1=A(L5)
      Y2=A(L5+1)
      Y3=A(L5-1)
  676 F5=TERPOL(Y1, Y2, Y3, TDIF5, T1, T2, T3)
      IF(F5.LT.0.D0) F5=0.D0
      IF(F5.GT.1.D0) F5=1.D0
      GO TO 651
  650 F5=1.D0
  651 IF(TDIF6.GE.CDF(J6,12)) GO TO 655
  652 IF(TDIF6.GT.CDF(J6,L6)) GO TO 656
      L6=L6-1
      IF(1-L6)652,652,641
  655 F6=1.D0
      GO TO 659
  656 IF (L6 .NE. 1) GO TO 677
      T1=CDF(J6,1)
      T2=CDF(J6,2)
      T3=CDF(J6,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 678
  677 T1=CDF(J6,L6)
      T2=CDF(J6,L6+1)
      T3=CDF(J6,L6-1)
```

```
      Y1=A(L6)
      Y2=A(L6+1)
      Y3=A(L6-1)
  678 F6=TERPOL(Y1, Y2, Y3, TDIF6, T1, T2, T3)
      IF(F6.LT.0.D0)F6=0.D0
      IF(F6.GT.1.D0) F6=1.D0
  659 FOT1=FOT1+F5*F6
      GO TO 640
  660 DENS(J)= FOT*0.1D-2
      IF(DENS(J).LT.0.D0) DENS(J)=0.D0
      IF(DENS(J).GE.1.D0) GO TO 680
  670 CONTINUE
  680 DO 681 JJ=J,10
      DENS(JJ)=1.D0
  681 CONTINUE
      S(J1,2)=S(I,2)
      S(J1,3)=S(J6,3)
      CALL INVSTR(J1,DELTAT)
      RETURN
      END
      SUBROUTINE CWIRED(I,J1,J2,J3,J4,J9,J10,J11)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      DO 710 J=1,2
      L=1+(J-1)*11
      T1=CDF(I,L)+CDF(J1,L)
      T2=CDF(I,L)+CDF(J2,L)+CDF(J4,L)
      T3=CDF(J10,L)+CDF(J11,L)
      T4=CDF(J9,L)+CDF(J10,L)+CDF(J4,L)
      T5=CDF(J3,L)+CDF(J4,L)
      IF(T1.GE.T2.AND.T1.GE.T3.AND.T1.GE.T4.AND.T1.GE.T5) GO TO 701
      IF(T2.GE.T1.AND.T2.GE.T3.AND.T2.GE.T4.AND.T2.GE.T5)GO TO 702
      IF(T3.GE.T1.AND.T3.GE.T2.AND.T3.GE.T4.AND.T3.GE.T5) GO TO 703
      IF(T4.GE.T1.AND.T4.GE.T2.AND.T4.GE.T3.AND.T4.GE.T5) GO TO 704
      TI(J)=T5
      GO TO 710
  701 TI(J)=T1
      GO TO 710
  702 TI(J)=T2
      GO TO 710
  703 TI(J)=T3
      GO TO 710
  704 TI(J)=T4
  710 CONTINUE
      DENS(1)=0.D0
      DENS(11)=1.0D0
      DELTAT=(TI(2)-TI(1))/10.0D0
      DO 790 J=2,10
      T=TI(1) + DFLOAT(J-1)*DELTAT
      K4=1
      FOT=0.D0
      L3=11
  715 K4=K4+1
      IF(K4.GE.12) GO TO 785
      TDIF3=T-CDF(J4,K4)
      IF(TDIF3.GE.CDF(J3,12)) GO TO 720
  716 IF(TDIF3.GT.CDF(J3,L3)) GO TO 717
      L3=L3-1
      IF(1-L3)716,716,785
```

```
717 IF (L3 .NE. 1) GO TO 791
    T1=CDF(J3,1)
    T2=CDF(J3,2)
    T3=CDF(J3,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 792
791 T1=CDF(J3,L3)
    T2=CDF(J3,L3+1)
    T3=CDF(J3,L3-1)
    Y1=A(L3)
    Y2=A(L3+1)
    Y3=A(L3-1)
792 F3=TERPOL(Y1, Y2, Y3, TDIF3, T1, T2, T3)
    IF(F3 .LT.0.D0) F3=0.D0
    IF(F3.GT.1.D0) F3=1.D0
    GO TO 721
720 F3=1.D0
721 KI=1
    FOT1=0.D0
    L2=11
    L1=11
725 KI=KI+1
    IF(KI.LT.12) GO TO 730
726 FOT=FOT+FOT1*F3
    GO TO 715
730 TDIF2=TDIF3-CDF(I,KI)
    TDIF1=T-CDF(I,KI)
    IF(TDIF2.GE.CDF(J2,12)) GO TO 732
731 IF(TDIF2.GT.CDF(J2,L2)) GO TO 733
    L2=L2-1
    IF(1-L2)731,731,726
732 F2=1.D0
    GO TO 735
733 IF (L2 .NE. 1) GO TO 793
    T1=CDF(J2,1)
    T2=CDF(J2,2)
    T3=CDF(J2,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 974
793 T1=CDF(J2,L2)
    T2=CDF(J2,L2+1)
    T3=CDF(J2,L2-1)
    Y1=A(L2)
    Y2=A(L2+1)
    Y3=A(L2-1)
974 F2=TERPOL(Y1, Y2, Y3, TDIF2, T1, T2, T3)
    IF(F2 .LT.0.D0) F2=0.D0
    IF(F2 .GT.1.D0) F2=1.D0
735 IF(TDIF1.GE.CDF(J1,12)) GO TO 737
736 IF(TDIF1.GT.CDF(J1,L1)) GO TO 738
    L1=L1-1
    IF(1-L1)736,736,726
737 F1=1.D0
    GO TO 740
738 IF (L1 .NE. 1) GO TO 795
    T1=CDF(J1,1)
    T2=CDF(J1,2)
```

```
      T3=CDF(J1,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 796
  795 T1=CDF(J1,L1)
      T2=CDF(J1,L1+1)
      T3=CDF(J1,L1-1)
      Y1=A(L1)
      Y2=A(L1+1)
      Y3=A(L1-1)
  796 F1=TERPOL(Y1, Y2, Y3, TDIF1, T1, T2, T3)
      IF(F1.LT.0.D0) F1=0.D0
      IF(F1.GT.1.D0) F1=1.D0
  740 K10=1
      FCT2=0.D0
      L9=11
      L11=11
  741 K10=K10+1
      IF(K10.LT.12) GO TO 743
  742 FCT1=FCT1+F2*F1*FCT2
      GO TO 725
  743 TDIF9=TDIF3-CDF(J10,K10)
      TDIF11=T-CDF(J10,K10)
      IF(TDIF9.GE.CDF(J9,12)) GO TO 747
  744 IF(TDIF9.GT.CDF(J9,L9)) GO TO 748
      L9=L9-1
      IF(1-L9)744,744,742
  747 F9=1.D0
      GO TO 750
  748 IF (L9 .NE. 1) GO TO 797
      T1=CDF(J9,1)
      T2=CDF(J9,2)
      T3=CDF(J9,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 798
  797 T1=CDF(J9,L9)
      T2=CDF(J9,L9+1)
      T3=CDF(J9,L9-1)
      Y1=A(L9)
      Y2=A(L9+1)
      Y3=A(L9-1)
  798 F9=TERPOL(Y1, Y2, Y3, TDIF9, T1, T2, T3)
      IF(F9.LT.0.D0) F9=0.D0
      IF(F9.GT.1.D0) F9=1.D0
  750 IF(TDIF11.GE.CDF(J11,12)) GO TO 755
  751 IF(TDIF11.GT.CDF(J11,L11)) GO TO 756
      L11=L11-1
      IF(1-L11)751,751,742
  755 F11=1.D0
      GO TO 760
  756 IF (L11.NE. 1) GO TO 799
      T1=CDF(J11,1)
      T2=CDF(J11,2)
      T3=CDF(J11,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 800
```

```
799 T1=CDF(J11,L11)
    T2=CDF(J11,L11+1)
    T3=CDF(J11,L11-1)
    Y1=A(L11)
    Y2=A(L11+1)
    Y3=A(L11-1)
800  F11=TERPOL(Y1, Y2, Y3, TDIF11, T1, T2, T3)
    IF(F11.LT.0.D0) F11=0.D0
    IF(F11.GT.1.D0) F11=1.D0
760 FOT2=FOT2+F9*F11
    GO TO 741
785 DENS(J)= FOT*0.1D-2
    IF(DENS(J).LT.0.D0) DENS(J)=0.D0
    IF(DENS(J).GE.1.D0) GO TO 780
790 CONTINUE
780 DO 781 JJ=J,10
    DENS(JJ)=1.D0
781 CONTINUE
    S(J1,2)=S(I,2)
    CALL INVSTR(J1,DELTAT)
    RETURN
    END
    SUBROUTINE DW2RED(I,J1,J2,J3,J4,J8,J10,J11)
    IMPLICIT REAL*8(A-H,O-R,T-Z)
    COMMON /BLKA/S,CDF,A,DENS,TI
    INTEGER S
    DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
    DO 810 J=1,2
    L=1+(J-1)*11
    T1=CDF(I,L)+CDF(J1,L)
    T2=CDF(J2,L)+CDF(I,L)+CDF(J4,L)
    T3=CDF(I,L)+CDF(J2,L)+CDF(J8,L)+CDF(J11,L)
    T4=CDF(J3,L)+CDF(J4,L)
    T5=CDF(J3,L)+CDF(J8,L)+CDF(J11,L)
    T6=CDF(J10,L)+CDF(J11,L)
    IF(T1.GE.T2.AND.T1.GE.T3.AND.T1.GE.T4.AND.T1.GE.T5.AND.T1.GE.T6)
   1GO TO 801
    IF(T2.GE.T1.AND.T2.GE.T3.AND.T2.GE.T4.AND.T2.GE.T5.AND.T2.GE.T6)
   1GO TO 802
    IF(T3.GE.T1.AND.T3.GE.T2.AND.T3.GE.T4.AND.T3.GE.T5.AND.T3.GE.T6)
   1GO TO 803
    IF(T4.GE.T1.AND.T4.GE.T2.AND.T4.GE.T3.AND.T4.GE.T5.AND.T4.GE.T6)
   1GO TO 804
    IF(T5.GE.T1.AND.T5.GE.T2.AND.T5.GE.T3.AND.T5.GE.T4.AND.T5.GE.T6)
   1GO TO 805
    TI(J)=T6
    GO TO 810
801 TI(J)=T1
    GO TO 810
802 TI(J)=T2
    GO TO 810
803 TI(J)=T3
    GO TO 810
804 TI(J)=T4
    GO TO 810
805 TI(J)=T5
810 CONTINUE
    DENS(1)=0.D0
    DENS(11)=1.0D0
    DELTAT=(TI(2)-TI(1))/10.0D0
    DO 890 J=2,10
```

```
      T=TI(1) + DFLOAT(J-1)*DELTAT
      K11=1
      L10=11
      FOT=0.D0
  815 K11=K11+1
      IF(K11.GE.12) GO TO 885
      TDIF10=T-CDF(J11,K11)
      IF(TDIF10.GE.CDF(J10,12)) GO TO 817
  816 IF(TDIF10.GT.CDF(J10,L10)) GO TO 818
      L10=L10-1
      IF(1-L10)816,816,885
  817 F10=1.D0
      GO TC 820
  818 IF (L10.NE. 1) GO TO 791
      T1=CDF(J10,1)
      T2=CDF(J10,2)
      T3=CDF(J10,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 792
  791 T1=CCF(J10,L10)
      T2=CDF(J1C,L10+1)
      T3=CCF(J10,L10-1)
      Y1=A(L10)
      Y2=A(L10+1)
      Y3=A(L10-1)
  792  F10=TERPOL(Y1, Y2, Y3, TDIF10, T1, T2, T3)
      IF(F10.LT.0.D0)F10=0.D0
      IF(F10.GT.1.D0) F10=1.D0
  820 KI=1
      FOT3=0.D0
      L1=11
  821 KI=KI+1
      IF(KI.LT.12) GO TO 823
  822 FCT=FCT+F10*FOT3
      GC TO 815
  823 TDIF1=T-CDF(I,KI)
      IF(TDIF1.GE.CDF(J1,12)) GO TO 825
  824 IF(TDIF1.GT.CDF(J1,L1)) GO TO 826
      L1=L1-1
      IF(1-L1)824,824,822
  825 F1=1.D0
      GO TO 830
  826 IF (L1 .NE. 1) GO TO 893
      T1=CDF(J1,1)
      T2=CDF(J1,2)
      T3=CDF(J1,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 894
  893 T1=CCF(J1,L1)
      T2=CDF(J1,L1+1)
      T3=CDF(J1,L1-1)
      Y1=A(L1)
      Y2=A(L1+1)
      Y3=A(L1-1)
  894 F1=TERPOL(Y1, Y2, Y3, TDIF1, T1, T2, T3)
      IF(F1.LT.0.D0) F1=0.D0
      IF(F1.GT.1.D0) F1=1.D0
```

```
830 K8=1
    FOT2=0.D0
831 K8=K8+1
    IF(K8.LT.12) GO TO 832
    FOT3=FOT3+F1*FOT2
    GO TO 821
832 K4=1
    FOT1=0.D0
    L2=11
    L3=11
833 K4=K4+1
    IF(K4.GE.12) GO TO 840
    T1=T-CDF(J4,K4)
    T2=TDIF10-CDF(J8,K8)
    IF(T1-T2)834,835,835
834 TDIF3=T1
    GO TO 836
835 TDIF3=T2
836 TDIF2=TDIF3-CDF(1,K1)
    IF(TDIF2.GE.CDF(J2,12)) GO TO 838
837 IF(TDIF2.GT.CDF(J2,L2)) GO TO 839
    L2=L2-1
    IF(1-L2)837,837,840
840 FOT2=FOT2+FOT1
    GO TO 831
838 F2=1.D0
    GO TO 845
839 IF (L2 .NE. 1) GO TO 895
    T1=CDF(J2,1)
    T2=CDF(J2,2)
    T3=CDF(J2,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 896
895 T1=CDF(J2,L2)
    T2=CDF(J2,L2+1)
    T3=CDF(J2,L2-1)
    Y1=A(L2)
    Y2=A(L2+1)
    Y3=A(L2-1)
896 F2=TERPOL(Y1, Y2, Y3, TDIF2, T1, T2, T3)
    IF(F2 .LT.0.D0) F2=0.D0
    IF(F2 .GT.1.D0) F2=1.D0
845 IF(TDIF3.GE.CDF(J3,12)) GO TO 850
846 IF(TDIF3.GT.CDF(J3,L3)) GO TO 851
    L3=L3-1
    IF(1-L3)846,846,840
850 F3=1.D0
    GO TO 860
851 IF (L3 .NE. 1) GO TO 897
    T1=CDF(J3,1)
    T2=CDF(J3,2)
    T3=CDF(J3,3)
    Y1=0.D0
    Y2=0.05D0
    Y3=0.15D0
    GO TO 898
897 T1=CDF(J3,L3)
    T2=CDF(J3,L3+1)
    T3=CDF(J3,L3-1)
```

```
      Y1=A(L3)
      Y2=A(L3+1)
      Y3=A(L3-1)
  858 F3=TERPOL(Y1, Y2, Y3, TDIF3, T1, T2, T3)
      IF(F3 .LT.0.D0) F3=0.D0
      IF(F3.GT.1.D0) F3=1.D0
  860 FOT1=FOT1+F3*F2
      GO TO 833
  885 DENS(J)= FOT*0.1D-3
      IF(DENS(J).LT.0.D0) DENS(J)=0.D0
      IF(DENS(J).GE.1.D0) GO TO 880
  890 CONTINUE
  880 DO 881 JJ=J,10
      DENS(JJ)=1.D0
  881 CONTINUE
      S(J1,2)=S(I,2)
      CALL INVSTR(J1,DELTAT)
      RETURN
      END
      SUBROUTINE WBRED(I,J1,J2,J3,J4)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
C     *     *     *     *     *     *     *     *     *     *     *
C
C
      DO 910 J=1,2
      L=1+(J-1)*11
      T1=CDF(I,L)+CDF(J1,L)
      T2=CDF(I,L)+CDF(J2,L)+CDF(J4,L)
      T3=CDF(J3,L)+CDF(J4,L)
      IF(T1-T2) 901,901,902
  901 IF(T2-T3) 903,903,904
  902 IF(T1-T3)903,903,905
  903 TI(J)=T3
      GO TO 910
  904 TI(J)=T2
      GO TO 910
  905 TI(J)=T1
  910 CONTINUE
      DENS(1)=0.D0
      DENS(11)=1.0D0
      DELTAT=(TI(2)-TI(1))/10.0D0
      DO 990 J=2,10
      T=TI(1)+DFLOAT(J-1)*DELTAT
      K4=1
      FOT=0.D0
      L3=11
  915 K4=K4+1
      IF(K4.GE.12) GO TO 985
      TDIF3=T-CDF(J4,K4)
      IF(TDIF3.GE.CDF(J3,12)) GO TO 918
  916 IF(TDIF3.GT.CDF(J3,L3)) GO TO 917
      L3=L3-1
      IF(1-L3)916,916,985
C
C
C
C     *     *     *     *     *     *     *     *     *     *     *
  917 IF (L3 .NE. 1) GO TO 951
```

```
      T1=CDF(J3,1)
      T2=CDF(J3,2)
      T3=CDF(J3,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 952
  951 T1=CDF(J3,L3)
      T2=CDF(J3,L3+1)
      T3=CDF(J3,L3-1)
      Y1=A(L3)
      Y2=A(L3+1)
      Y3=A(L3-1)
  952 F3=TERPOL(Y1, Y2, Y3, TDIF3, T1, T2, T3)
      IF(F3 .LT.0.D0) F3=0.D0
      IF(F3.GT.1.D0) F3=1.D0
C     *    *    *    *    *    *    *    *    *    *    *
C
C
C
      GO TO 920
  918 F3=1.D0
  920 KI=1
      FOT1=0.D0
      L2=11
      L1=11
  921 KI=KI+1
      IF(KI.LT.12) GO TO 925
  922 FOT=FOT+FOT1*F3
      GO TO 915
  925 TDIF2=TDIF3-CDF(I,KI)
      TDIF1=T-CDF(I,KI)
      IF(TDIF2.GE.CDF(J2,12)) GOTO 928
  926 IF(TDIF2.GT.CDF(J2,L2)) GO TO 927
      L2=L2-1
      IF(1-L2)926,926,922
C
C
C
C     *    *    *    *    *    *    *    *    *    *    *
  927 IF (L2 .NE. 1) GO TO 961
      T1=CDF(J2,1)
      T2=CDF(J2,2)
      T3=CDF(J2,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 962
  961 T1=CDF(J2,L2)
      T2=CDF(J2,L2+1)
      T3=CDF(J2,L2-1)
      Y1=A(L2)
      Y2=A(L2+1)
      Y3=A(L2-1)
  962 F2=TERPOL(Y1, Y2, Y3, TDIF2, T1, T2, T3)
      IF(F2 .LT.0.D0) F2=0.D0
      IF(F2 .GT.1.D0) F2=1.D0
C     *    *    *    *    *    *    *    *    *    *    *
C
C
C
```

```
      GO TO 930
  928 F2=1.D0
  930 IF(TDIF1.GE.CDF(J1,12)) GO TO 933
  931 IF(TDIF1.GT.CDF(J1,L1)) GO TO 932
      L1=L1-1
      IF(1-L1)931,931,922
C
C
C
C     *     *     *     *     *     *     *     *     *     *     *
  932 IF (L1 .NE. 1) GO TO 971
      T1=CDF(J1,1)
      T2=CDF(J1,2)
      T3=CDF(J1,3)
      Y1=0.D0
      Y2=0.05D0
      Y3=0.15D0
      GO TO 972
  971 T1=CDF(J1,L1)
      T2=CDF(J1,L1+1)
      T3=CDF(J1,L1-1)
      Y1=A(L1)
      Y2=A(L1+1)
      Y3=A(L1-1)
  972 F1=TERPOL(Y1, Y2, Y3, TDIF1, T1, T2, T3)
      IF(F1.LT.0.D0) F1=0.D0
      IF(F1.GT.1.D0) F1=1.D0
C     *     *     *     *     *     *     *     *     *     *     *
C
C
C
      GO TO 940
  933 F1=1.D0
  940 FCT1=FCT1+F1*F2
      GO TO 921
  985 DENS(J)=FCT*0.1D-1
  990 CONTINUE
      S(J1,2)=S(I,2)
      CALL INVSTR(J1,DELTAT)
      RETURN
      END
      FUNCTION TERPOL(X1,X2,X3,Y,Y1,Y2,Y3)
      IMPLICIT REAL*8(A-H,O-Z)
      TERPOL=X1 + ((X2-X1)*(Y-Y1)/(Y2-Y1)) + ((X3-X2)/(Y3-Y2)-(X2-X1)/
     *(Y2-Y1))*((Y-Y1)*(Y-Y2)/(Y3-Y1))
      RETURN
      END
      SUBROUTINE INVSTR(I,DELTAT)
C
C     CDF INVERSE STORAGE ROUTINE
C
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKA/S,CDF,A,DENS,TI
      INTEGER S
      DIMENSION S(1000,4),CDF(1000,12),A(12),DENS(400),TI(2)
      TEMPO=TI(1)
      MIN=1
      DO 2263 IA=2,10
      IF (DENS(IA).GE.1.D0) GO TO 2273
      IF (DENS(IA).GT.0.D0) GO TO 2263
      MIN=IA
```

```
      TI(1)=TEMPO+DFLOAT(IA-1)*DELTAT
 2263 CONTINUE
      MAX=11
      GO TO 251
 2273 MAX=IA
      TI(2)=TEMPO+DFLOAT(IA-1)*DELTAT
  251 DENS(MIN)=0.D0
      DENS(MAX)=1.D0
      JA=0
      DO 2283 KA=MIN,MAX
      JA=JA+1
      DENS(JA)=DENS(KA)
 2283 CONTINUE
      DO 2293 LA=JA,11
 2293 DENS(LA)=1.D0
  252 K=1
      CDF(I,1)=TI(1)
      CDF(I,12)=TI(2)
      DO 260 M=2,6
  255 IF(A(M).GT.DENS(K).AND.A(M).LE.DENS(K+1)) GO TO 256
      K=K+1
      GO TO 255
  256 Y=A(M)
  257 Y1=DENS(K)
      Y2=DENS(K+1)
      Y3=DENS(K+2)
      T1=TI(1) + DFLOAT(K-1)*DELTAT
      D1=Y2-Y1
      D2=Y3-Y2
      D3=Y3-Y1
      DY1=Y-Y1
      DY2=Y-Y2
      CTEMP   =T1+(DY1*DELTAT/D1)+(DY1*DY2*DELTAT*(D1-D2)/(D1*D2*D3))
      IF(CTEMP.LT.T1)CTEMP=T1
      IF(CTEMP.GT.T1+2*DELTAT)CTEMP=T1+2*DELTAT
 260  CDF(I,M)=CTEMP
      DO 270 M=7,11
  265 IF(A(M).GT.DENS(K).AND.A(M).LE.DENS(K+1)) GO TO 266
      K=K+1
      GO TO 265
  266 Y=A(M)
      Y1=DENS(K+1)
      Y2=DENS(K)
      Y3=DENS(K-1)
      T1=TI(1) + DFLOAT(K)*DELTAT
      D1=Y2-Y1
      D2=Y3-Y2
      D3=Y3-Y1
      DY1=Y-Y1
      DY2=Y-Y2
      CTEMP   =T1-(DY1*DELTAT/D1)-(DY1*DY2*DELTAT*(D1-D2)/(D1*D2*D3))
      IF(CTEMP.LT.T1)CTEMP=T1
      IF(CTEMP.GT.T1+2*DELTAT)CTEMP=T1+2*DELTAT
 270  CDF(I,M)=CTEMP
      RETURN
      END
      SUBROUTINE XDENS(MM,I,I500,J200)
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      COMMON /BLKB/TIME,V
      DIMENSION TIME(4),V(12)
      GO TO (30,31,50,60),MM
```

```
50     CALL XRECT
       GO TO 70
60     DO 61 M=1,12
61     Y(M)=TIME(1)
       TIME(3)=0.D0
       GO TO 70
   31  IF(TIME(4))32,32,33
32     TIME(3)=(TIME(3)-TIME(1))/6.0
   33  TIME(1)=TIME(2)
34     CALL NORMAL(J200)
       GO TO 70
30     CALL XBETA(I500)
70     CONTINUE
       RETURN
       END
       SUBROUTINE XRECT
C      SUBROUTINE TO CALCULATE THE INVERSE FUNCTION OF A RECTANGULAR DIST
C      CALLING ARGUMENTS
C        A IS THE MINIMUM VALUE OF THE VARIABLE
C        B IS THE MAXIMUM VALUE OF THE VARIABLE
C        Y IS THE ARRAY OF THE INVERSE FUNCTION OF THE DISTRIBUTION
C      THE MEAN IS RETURNED IN A
C      THE STANDARD DEVIATION IS RETURNED IN B
C
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON /BLKB/TIME,Y
       DIMENSION TIME(4),Y(12)
       A=TIME(1)
       B=TIME(3)
       DELTA=(TIME(3)-TIME(1))/10.D0
       Y(1)=A
       Y( 12)=B
       Y(2)=Y(1)+DELTA*0.5D0
       DO 10 J=3,11
   10  Y(J)=Y(J-1)+DELTA
       XMU=(A+B)/2.D0
       SIGMA=(B-A)/3.4641D0
       TIME(1)=XMU
       TIME(3)=SIGMA
        RETURN
       END
       SUBROUTINE XBETA(IG)
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON /BLKB/TIME,Y
       DIMENSION TIME(4),Y(12)
       TERP  (X1,X2,Y,Y1,Y2)=X1+((X2-X1)*(Y-Y1)/(Y2-Y1))
C TERP   IS A ARITHMETIC STATEMENT FUNCTION TO DO LINEAR INTERPOLATION.
C IF X IS THE VALUE REQUIRED WHICH LIES BETWEEN THE VALUES  OF X1 AND X2
C IN A TABLE AND Y1 AND Y2 ARE THECORRESPONDING VALUES WITH Y THE VALUE
C CORRESPONDING TO THE VALUE OF X REQUIRED.
C SUB PROGRAM TO CALCULATE THE CUMULATIVE INVERSE FUNCTION OF THE BETA
C                                DENSITY FUNCTION
C
C
C
C        CALLING ARGUMENTS.
C          1. BETA # BETA OF THE DENSITY IF KNOWN. ENTER -1 IF IT IS
C             DESIRED TO CALCULATE THESE POINTS BASED ON THE THREE PERT
C             TIME ESTIMATES  XSEE SUBROUTINE ALBET<.
C          2. A # MINIMUM TIME ESTIMATE.
C          3. XM # ALPHA OF THE DENSITY OR THE MOST LIKLEY TIME ESTIMATE.
C          4. B # MAXIMUM TIME ESTIMATE.
```

```
C         5. IG # SEE GAMMA SUBROUTINE.
C         6. Y # ARRAY 101 X 1 IN WHICH THE VALUES ARE CALCULATED
C         7.  SIGMA IS THE STANDARD DEVIATION OF THE DENSITY
C      THE MEAN OF THE DENSITY IS RETURNED IN A.
C      THE STANDARD DEVIATION OF THE DENSITY IS RETURNED IN TIMEX3<
C
       A=TIME(1)
       ALPHA=TIME(2)
       B=TIME(3)
       BETA=TIME(4)
       Y(1)=A
       Y(12)=B
    18 FALPHA=ALPHA+1.D0
       XM=ALPHA
       FBETA=BETA + 1.D0
       SUM=ALPHA + BETA + 1.D0
       FSUM= SUM + 1.D0
       SIGMA=(FALPHA*FBETA*(B-A)**2)/(((FSUM)**2)*(FSUM+1.D0))
       SIGMA=DSQRT(SIGMA)
       U =  ((A*FBETA)+(B*FALPHA))/FSUM
       CALL GAMMA (FALPHA,IG)
       CALL GAMMA (FBETA,IG)
       CALL GAMMA  (FSUM,IG)
       C=B-A
       BT=0.D0
       DELTAT=C/250D0
       T3=0.D0
       FT=0.D0
       CONST=DLOG(FSUM)-DLOG(FALPHA)-DLOG(FBETA)-(SUM*DLOG(C))
       Z3=0.D0
       K=0
       DO 50 J=2,20,2
       D=DFLOAT(J-1)/20.0
    49 T1=T3
       K=K+1
       X=K
       XX=K*2
       AT=BT
       T3=X*DELTAT
       T2=(XX-1.D0)*T3/XX
       Z1=Z3
       Z2=CONST+DLOG(4.D0)+(ALPHA*DLOG(T2))+(BETA*DLOG(C-T2))
       Z2=DEXP(Z2)
       Z3=CONST+(ALPHA*DLOG(T3))+(BETA*DLOG(C-T3))
       Z3=DEXP(Z3)
       FT=FT+Z1+Z2+Z3
       BT=FT*DELTAT/6.D0
       IF(BT.GT.1.D0) BT=1.D0
       IF(D.GT.AT.AND.D.LE.BT)  GO TO 51
       GO TO 49
    51 NJ=(J/2)+1
       Y(NJ)=TERP  (T1,T3,D,AT,BT) + Y(1)
    50 CONTINUE
       TIME(1)=U
       TIME(2)=ALPHA
       TIME(3)=SIGMA
       TIME(4)=BETA
       RETURN
       END
       SUBROUTINE GAMMA (X,M)
C
```

```
C
C   SUBROUTINE TO COMPUTE THE GAMMA FUNCTION XSINGLE PRECISION<
C
C        CALLING ARGUMENTS
C               X IS THE NUMBER OF WHICH THE GAMMA IS TO BE COMPUTED
C               X IS A FLOATING NUMBER
C               M IS AN INTEGER NUMBER
C               M IS DEFINED AS FOLLOWS
C                    M MUST BE ZERO THE FIRST TIME THE SUBROUTINE IS CALLE
C
      IMPLICIT REAL*8(A-H,O-R,T-Z)
      DIMENSION Z(18)
      IF(M)  1, 1, 2
    1 B=1.D0
      D=2.D0
      M=1
      Z( 1)= 1.000000000D 00
      Z( 2)=  .5772156664D 00
      Z( 3)=- .6558780071D 00
      Z( 4)=- .0420026350D 00
      Z( 5)=  .1665386110D 00
      Z( 6)=- .0421977340D 00
      Z( 7)=- .0096219710D 00
      Z( 8)=  .0072189430D 00
      Z( 9)=- .0011651670D 00
      Z(10)=- .0002152410D 00
      Z(11)=  .0001280500D 00
      Z(12)=- .0000201340D 00
      Z(13)=- .0000012500D 00
      Z(14)=  .0000011330D 00
      Z(15)=- .0000002050D 00
      Z(16)=  .0000000060D 00
      Z(17)=  .0000000050D 00
      Z(18)=- .0000000010D 00
    2 C=1.D0
    3 IF(X) 99,99, 4
    4 IF(X-B)  7, 6, 5
    5 IF(D-X) 8, 6, 9
    6 G=1.D0
      GO TO 11
    7 C=C*B/X
      X=X+B
      GO TO 3
  8   X=X-B
      C=C*X
      GO TO 3
    9 E=0.D0
      DO 10 I=1,18
      T=I
   10 E=E+(Z(I)*(X**T))
      G=B/E
   11 X=C*G
   99 RETURN
      END
      SUBROUTINE NORMAL(J2)
C   SUBROUTINE TO CALCULATE THE INVERSE FUNCTION OF THE NORMAL DISTRIB.
C   CALLING ARGUMENTS.
C     1.  Y IS A FLOATING POINT ARRAY FOR OUTPUT OF THE CUMULATIVE INVERS
C         FUNCTION.
C     2.  U IS THE MEAN OF THE DENSITY.
C     3.  SIGMA IS THE STANDARD DEVIATION OF THE DENSITY.
```

```
C      4.  Z IS THE ARRAY IN WHICH THE VALUES OF THE FUNCTION ARE CALCULAT
C      5.  J2 IS A DUMMY VARIABLE WHICH MUST BE 0 OR 1 THE FIRST TIME THE
C          IS CALLED OR ANYTIME THE Z ARRAY IS USED AGAIN BY THE MAIN
C          PROGRAM.  WHEN J2 IS       MINUS THE Z TABLE IS NOT RECALCULATED
C                 THE PROGRAM RETURNS VALUES OF Y FOR THE FUNCTION
C       T # U &Z * SIGMA        FOR VALUES OF F%Y<#.01,.02, ....,.99, .1.0.
C     YX1< WILL CONTAIN U - 3 STANDARD DEVIATIONS.IF & OR.0 IF -. NEGATIVE
C     VALUES OF Y ARE NOT CALCULATED.  YX 22< WILL CONTAIN U & 3 STANDARD
C     DEVIATIONS.
        IMPLICIT REAL*8(A-H,O-Z)
        COMMON /BLKB/TIME,Y
        DIMENSION TIME(4),Y(12)
        DIMENSION Z(10)
        U=TIME(2)
        SIGMA=TIME(3)
        IF (J2) 5,10,10
     10 J2=-1
        Z(1)=-1.6449300
        Z(2)=-1.0364500
        Z(3)=-0.6744900
        Z(4)=-0.3853200
        Z(5)=-0.1256600
        Z( 6)=0.1256600
        Z( 7)=0.3853200
        Z( 8)=0.6744900
        Z( 9)=1.0364500
        Z(10)=1.6449300
    5   TZERO=U-(3.D0*SIGMA)
        IF (TZERO) 1,2,2
      1 Y(1)=0.D0
        GO TO 8
    2   Y(1)=TZERO
    8   DO 9 I=2,11
        Y(I)=Z(I-1)*SIGMA+U
        IF (Y(I)) 6,6,9
      6 Y(I)=0.D0
    9   CONTINUE
        Y(12)=U+3.D0*SIGMA
        RETURN
        END
        SUBROUTINE RD(J,A,N)
        DIMENSION A(N)
        READ(J) A
        RETURN
        END
```

6. DECOMP

```
C     DECOMPOSITION PROGRAM
C
C        DECOMP  IDENTIFIES PARALLEL SUBNETWORKS AND SERIES SUBNETWORKS
C
C
C        THE FOLLOWING IS AN ALPHABETICAL LISTING OF THE VARIABLES AND
C        ARRAYS THAT ARE USED IN THIS MAIN PROGRAM AND ITS SUBROUTINES
C
C
C        ARCS = THE NUMBER OF ARCS IN THE SUBNETWORK
C        BDNUM(I) = THE BUNDLE NUMBER TO WHICH NODE I IS ASSIGNED
C        CARC = AN ARRAY PUNCHED FOR USE IN PROGRAM: SYNTHESIS
C        CHECK = ARRAY USED TO STORE ARCS HAVING THE SOURCE AND THE
C               SINK AS THEIR ONLY NODES
C        CTNSUB = THE TEMPORARY NUMBER OF SUBNETWORKS FOUND IN THE
C                PREVIOUS STEP
C        HEAD(I)=TERMINATING NODE FOR ARC I IN THE RENUMBERED SUBNETWORK
C        INSTN = THE CURRENT INSTRUCTION NUMBER, PUNCHED FOR SYNTHESIS
C        LNODEN = THE LARGEST NODE NUMBER BEING READ IN
C        MAXND = THE LARGEST NODE NUMBER THAT HAS ALREADY BEEN ASSIGNED
C             AT LEAST TEMPORARILY TO A BUNDLE
C        NARCSS(I) = THE NUMBER OF ARCS IN SUBNETWORK I
C        NODES=TEMPORARY ARRAY USED TO ORDER THE NODE NUMBERS IN A SUB
C        NSUB = THE TOTAL OF SUBNETWORKS THUS FAR
C        NTARC = THE NUMBER OF ARCS IN THIS SUBNETWORK
C        NUMBD = THE NUMBER OF BUNDLES CREATED
C        NUMSUB=NUMBER OF MINIMUM SUBNETWORKS IDENTIFIED
C        PPOST = ARRAY OF PAST POSTS USED IN CUT SUBROUTINE
C        S(I) = THE STARTING NODE FOR ARC I
C        SINK   = THE NODE NUMBER CORRESPONDING TO THE SINK
C        SINKS(I) = THE SINK IN SUBNETWORK I
C        SOURC(I) = SOURCE NODE IN SUBNETWORK I
C        SOURCE = THE NODE NUMBER CORRESPONDING TO THE SOURCE
C        STEP = STAGE NUMBER
C        SUBNET(I,J) = THE ITH ARC IN THE JTH SUBNETWORK
C        SUMARC = THE CURRENT NUMBER OF ARCS IN SUBNETWORK NSUB
C        T(I) = THE TERMINATING NODE FOR ARC I
C        TAIL(I)=STARTING NODE FOR ARC I IN THE RENUMBERED SUBNETWORK
C        TARC = THE SUBNETWORK WITHOUT THE ARCS INVOLVING NODE K
C        TLNODEN = TEMPORARY LARGEST NODE NUMBER
C        TNSUB = THE NUMBER OF SUBNETWORKS CREATED IN THE CURRENT STAGE
C        TSNSUB = THE TEMPORARY SUBNETWORK BEING USED IN STAGE
C                SUBROUTINE
C        TSUBN = THE NUMBER OF THE SUBNETWORK CURRENTLY BEING CONSIDERED
C        TYPESN = THE TYPE OF SUBNETWORK BEING CONSIDERED:
C             1 = BUNDLE SUBNETWORK  2 = CUT SUBNETWORK
C
C        A NON-DECOMPOSABLE SUBNETWORK IS ONE THAT IS NOT COMPOSED OF
C        SMALLER SUBNETWORKS WHICH ARE CONNECTED IN EITHER SERIES OF
C        PARALLEL
C
C
C        FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C        MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                  PROJECT NETWORK
C        MSMAX = THE MAXIMUM NUMBER OF ACTIVITIES ALLOWED IN ANY ONE
C                  SUBNETWORK
C        NSUBMAX = THE MAXIMUM NUMBER OF SUBNETWORKS ALLOWED IN THE
C                  DECOMPOSITION PROCESS
C        NCINSMAX = THE MAXIMUM NUMBER OF INSTRUCTIONS GENERATED BY THE
```

```
C                DECOMPOSITION PROCESS
C         CURRENTLY, MMAX=1000; MSMAX=500; NSUBMAX=100; NOINSMAX=100
C
C      DIMENSION S(MMAX),T(MMAX),SUBNET(MMAX,NSUBMAX),SOURC(NSUBMAX),
C     *          NARCSS(NSUBMAX),CARD(NOINS,27)
C
C         IN SUBROUTINE NETIN THE ARRAY DIMENSIONS ARE:
C      DIMENSION FILE3(3*MMAX+5),IORG(MMAX),ITERM(MMAX),NUMS(MMAX)
C      EQUIVLAENCE (FILE3(MMAX+6),IORG(1)),(FILE3(2*MMAX+6),ITERM(1))
C
C         IN SUBROUTINE BUNDLE THE ARRAY DIMENSIONS ARE:
C      DIMENSION BDNUM(MMAX),CHECK(MMAX)
C
C         IN SUBROUTINE CUT THE ARRAY DIMENSIONS ARE:
C      DIMENSION ORIGIN(MMAX),POST(MMAX),TARC(MMAX),RCUT(NSUBMAX),
C     *          PPOST(MMAX)
C
C         IN SUBROUTINE PRESUB THE ARRAY DIMENSIONS ARE:
C      DIMENSION HEAD(MMAX),TAIL(MMAX),NODE(MMAX),FILE3(3*MSMAX+3),
C     *          NET(3,MSMAX)
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      COMMON /NEW/ CARD,INSTN
      COMMON/PREP/ NUMSUB
      INTEGER                    F3/11/,          F5/13/
      INTEGER CARD(100,27)
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      WRITE(6,100)
100   FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE DECO
     *MPOSITION PROGRAM: DECOMP'/1H0,132('*')/1X,132('*'))
      REWIND F3
      READ(F3)
      REWIND F5
      READ(F5)
C
C         INITIALIZE INSTN
C         INSTN = THE CURRENT INSTRUCTION NUMBER
C
6000  INSTN=0
C
C         INITIALIZE CARD
C
      DO 22200 I=1,100
      DO 22200 J=1,27
22200 CARD(I,J)=0
C
C         READ THE NETWORK IN
C
      CALL NETIN (LNODEN)
      REWIND F3
      READ(F3)
      STEP=1
      TYPESN=1
      TSUBN=1
      NUMSUB=0
      CALL BUNDLE (LNODEN,TSUBN)
C
C         PRINT OUT THIS STAGE OF THE BREAKUP
C
```

```
      CALL STAGE (TYPESN,TSUBN)
      CTNSUB=TNSUB
      GO TO 70
C
C        FIND THE NUMBER OF NEWLY CREATED SUBNETWORKS
C
10    CTNSUB=0
      TSUBN=NSUB-LOOP
      TYPESN=1
      DO 20 I=1,LOOP
C
C        FIND THE NEXT SUBNETWORK TO BE FURTHER SUBDIVIDED
C
      TSUBN=TSUBN+1
C
C        FIND THE LARGEST NODE NUMBER IN SUBNETWORK TSUBN
C
      CALL NODER (TLNDEN,TSUBN)
C
C        FIND THE BUNDLE SUBNETWORKS
C
      CALL BUNDLE (TLNDEN,TSUBN)
C
C        PRINT OUT THIS STAGE OF THE BREAKUP
C
      CALL STAGE (TYPESN,TSUBN)
C
C        IF THERE IS ONLY ONE BUNDLE FOUND IN SUBNETWORK TSUBN, WE ARE
C        FINISHED.  PRINT OUT ITS COMPONENT ARCS
C
      IF (TNSUB.EQ.1) CALL ENDSNT (TSUBN)
C
C        COUNT THE NEW NUMBER OF SUBNETWORKS CREATED
C
20    CTNSUB=TNSUB+CTNSUB
C
C        IF ALL SUBNETWORKS ARE IN THEIR SMALLEST FORM, WE ARE FINISHED
C
      IF (CTNSUB.EQ.0) GO TO 90
      GO TO 70
80    LOOP=CTNSUB
      CTNSUB=0
      TSUBN=NSUB-LOOP
C
C        FIND THE NUMBER OF NEWLY CREATED SUBNETWORKS
C
      TYPESN=2
      DO 30 I=1,LOOP
C
C        FIND THE NEXT SUBNETWORK TO BE FURTHER SUBDIVIDED
C
      TSUBN=TSUBN+1
      IF (TSUBN.EQ.1) GO TO 55
C
C        FIND THE LARGEST NODE NUMBER IN SUBNETWORK TSUBN
C
      CALL NODER (TLNDEN,TSUBN)
C
C        FIND THE CUT SUBNETWORKS
C
55    IF (TSUBN.EQ.1) TLNDEN=LNODEN
```

```
      CALL CUT (TSUBN,TLNDEN)
C
C        PRINT OUT THIS STAGE OF THE BREAKUP
C
      CALL STAGE (TYPESN,TSUBN)
C
C        IF THERE ARE NO CUTS FOUND IN SUBNETWORK TSUBN, WE ARE FINISHED
C        PRINT OUT ITS COMPONENT ARCS
C
      IF (TNSUB.EQ.1) CALL ENDSNT (TSUBN)
C
C        COUNT THE NEW NUMBER OF SUBNETWORKS CREATED
C
30    CTNSUB=TNSUB+CTNSUB
C
C        IF ALL SUBNETWORKS ARE IN THEIR SMALLEST FORM, WE ARE FINISHED
C
      IF (CTNSUB.EQ.0) GO TO 90
      GO TO 70
85    LOOP=CTNSUB
      GO TO 10
70    WRITE (6,900) STEP
900   FORMAT (1H1,5X,'STAGE',I3,' BREAKUP')
      STEP=STEP+1
C
C        LET'S GO BACK TO THE APPROPRIATE LOOP FOR THE NEXT STAGE
C
      GO TO (80,85),TYPESN
90    CONTINUE
C
C        STORE INFORMATION FOR SYNTHESIS PROGRAM
C
      CALL PUNSYN
      WRITE(6,7001) NUMSUB
7001  FORMAT(1H0,'              THE NUMBER OF NON-DECOMPOSABLE SUBNETWORKS I
     *S 'I3)
      WRITE (6,9000)
9000  FORMAT (1H1)
      STOP
      END
      SUBROUTINE NETIN (LNGDEN)
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      INTEGER*4 FILE3(3005)
      INTEGER*4 IORG(1000),ITERM(1000),NUMS(1000)
      EQUIVALENCE (FILE3(6),NUMS(1)),(FILE3(1006),IORG(1)),
     *(FILE3(2006),ITERM(1))
      INTEGER                        F3/11/
      READ(F3) FILE3
C
C        ZEROIZE SOURC ARRAY
C
      DO 20 I=1,50
20    SOURC(I)=0
C
C        READ IN THE INITIAL NETWORK LIMITS
C
      ARCS = FILE3(1)
      SOURCE = FILE3(3)
```

```
      SINK = FILE3(4)
      LNODEN = FILE3(5)
      WRITE (6,200)
200   FORMAT (1H0,5X,'INPUT STAGE')
      WRITE (6,210) ARCS,SOURCE,SINK,LNODEN
210   FORMAT(1H0,/6X,'THE SIMPLIFIED NETWORK HAS',12X,I4,' ARCS',//6X,
     *'THE SOURCE IS NODE NUMBER',11X,I3,//6X,'THE SINK IS NODE NUMBER',
     *13X,I3,//6X,'THE LARGEST NODE IS NODE NUMBER',5X,I3)
      WRITE (6,220)
220   FORMAT(1H0,5X,'THE SIMPLIFIED NETWORK AS READ IN IS:',///6X,
     *'ARC NUMBER',5X,'ORIGIN NODE',5X,'TERMINAL NODE')
C
C        READ IN EACH ARC AND ITS STARTING AND TERMINATING NODES
C        S = THE NODE NUMBER FOR THE START OF AN ARC
C        T = THE TERMINAL NODE OF AN ARC
C
      DO 10 J=1,ARCS
      I=NUMS(J)
      S(I)=IORG(J)
      T(I)=ITERM(J)
      WRITE(6,240)I,S(I),T(I)
240   FORMAT (1H ,8X,I3,13X,I3,14X,I3)
C
C        CREATE THE FIRST SUBNETWORK
C
10    SUBNET(J,1)=I
      SOURC(1)=SOURCE
      SINKS(1)=SINK
      NARCSS(1)=ARCS
      NSUB=1
      RETURN
      END
      SUBROUTINE NODER (TLNDEN,TSUBN)
C
C        FINDS LARGEST NODE NUMBER IN THE SUBNETWORK TSUBN
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      ARCS=NARCSS(TSUBN)
      TLNDEN=0
      DO 20 J=1,ARCS
      A=SUBNET(J,TSUBN)
      M=S(A)
      N=T(A)
      MAXND=N
      IF (M.GT.N) MAXND=M
      IF (MAXND.GT.TLNDEN) TLNDEN=MAXND
20    CONTINUE
      RETURN
      END
      SUBROUTINE ENDSNT (TSUBN)
C
C        PRINTS SMALLEST BREAKDOWN OF SUBNETWORK TSUBN
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      SOURCE=SOURC(TSUBN)
```

```
      SINK=SINKS(TSUBN)
      WRITE (6,100) TSUBN
100   FORMAT (1H0,//16X,'SUBNETWORK ',I3,' IS A NON-DECOMPOSABLE NETWORK
     *',/16X,'IT IS COMPOSED OF:')
      WRITE (6,200) SOURCE,SINK
200   FORMAT (1H0,19X,'SOURCE NODE = ',I3,//20X,'SINK NODE   = ',I3)
      M=NARCSS(TSUBN)
      WRITE(6,500)
500   FORMAT(1H0,15X,'THE SUBNETWORK AS IT IS READ IN:')
      WRITE (6,400)
400   FORMAT (1H0,19X,'ARC',2X,'S(ARC)',2X,'T(ARC)')
      DO 10 I=1,M
      N=SUBNET(I,TSUBN)
10    WRITE (6,300) N,S(N),T(N)
300   FORMAT (1H ,19X,I3,3X,I3,5X,I3)
C
C         SET TNSUB=0 SO THAT THE REMAINING NUMBER OF SUBNETWORKS DOESN'T
C         INCLUDE THIS MINIMUM SUBNETWORK
C
      TNSUB=0
C
C         PREPARE THE MINIMUM SUBNETWORK FOR SUBNETWORK ANALYSIS
C
      CALL PRESUB(TSUBN)
      RETURN
      END
      SUBROUTINE STAGE (TYPESN,TSUBN)
C
C         PRINTS OUT THE CURRENT STAGE OF BREAKUP
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      COMMON /NEW/ CARD,INSTN
      INTEGER CARD(100,27)
C
C         TNSUB=THE NUMBER OF NEW SUBNETWORKS RESULTING FROM THE BREAKUP
C         IN THIS STAGE
C
C         TNSUB=1 IMPLIES NO BREAKUP OCCURRED IN THIS STAGE
C
      IF (TNSUB.EQ.1) GO TO 600
      WRITE (6,100) TSUBN
100   FORMAT (1H0,/6X,'SUBNETWORK ',I3,' IS COMPOSED OF SUBNETWORKS: ')
      TSNSUB=NSUB-TNSUB
      M=TSNSUB+1
      WRITE (6,300)(I,I=M,NSUB)
300   FORMAT (1H0,5X,I3,20(',',I3))
C
C         PREPARE INFORMATION FOR SYNTHESIS PROGRAM
C
      INSTN=INSTN+1
      ISORP=0
      IF(TYPESN.EQ.1) ISORP=1
      CARD(INSTN,1)=INSTN
      CARD(INSTN,2)=TSUBN
      CARD(INSTN,3)=ISORP
      DO 2250 I=M,NSUB
      K=I-M+4
 2250 CARD(INSTN,K)=I
```

```
      IF (TYPESN.EQ.1) GO TO 60
      WRITE (6,400)
400   FORMAT (1H0,5X,'IN SERIES')
      GO TO 600
60    WRITE (6,500)
500   FORMAT (1H0,5X,'IN PARALLEL')
600   RETURN
      END
      SUBROUTINE BUNDLE (LNODEN,TSUBN)
C
C        BUNDLE IDENTIFIES PARALLEL SUBNETWORKS CONNECTING DESIGNATED
C        SOURCE AND SINK
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      INTEGER*2 BDNUM(1000),CHECK(1000)
C
C        GROUP NODES INTO BUNDLES
C
       SOURCE=SOURC(TSUBN)
       SINK=SINKS(TSUBN)
      NUMBD=1
C
C        ZEROIZE THE BDNUM ARRAY
C
      DO 10 I=1,LNODEN
   10 BDNUM(I)=0
      K=SUBNET(1,TSUBN)
      M=S(K)
      N=T(K)
      BDNUM(M)=1
      BDNUM(N)=1
      MAXND=N
      IF (M.GT.N) MAXND=M
      ARCS=NARCSS(TSUBN)
      IF (ARCS.EQ.1) GO TO 515
      DO 1 K=2,ARCS
      I=SUBNET(K,TSUBN)
      BDNUM(SOURCE)=0
      BDNUM(SINK)=0
      M=S(I)
      N=T(I)
      IF(M.GT.MAXND) MAXND=M
      IF(N.GT.MAXND) MAXND=N
C
C        IF AT LEAST 1 NODE ON THE ARC HAS NOT BEEN ASSIGNED TO A BUNDLE
C        GO TO 2
C
      IF(BDNUM(M).EQ.0) GO TO 2
C
C        IF ONLY THE TERMINAL NODE ON THE ARC HAS NOT BEEN ASSIGNED TO A
C        BUNDLE, GO TO 3
C
      IF(BDNUM(N).EQ.0) GO TO 3
C
C        IF BOTH NODES ON THE ARC HAVE BEEN ASSIGNED TO THE SAME BUNDLE
C        EVERYTHING IS OKAY, GO TRY ANOTHER ARC
C
      IF(BDNUM(N).EQ.BDNUM(M)) GO TO 1
```

```
C
C        IF THE NODES ON THE ARC ARE ASSIGNED TO DIFFERENT BUNDLES,
C        THEN THESE TWO BUNDLES SHOULD BE POOLED
C
      IF(BDNUM(N).LT.BDNUM(M)) GO TO 6
C
C        POOL BUNDLES
C        THE BUNDLE WITH THE LARGER BUNDLE NUMBER IS POOLED INTO THE
C        BUNDLE WITH THE SMALLER BUNDLE NUMBER
C        THE BUNDLE NUMBERS OF ALL BUNDLES ARE ALL ADJUSTED
C
      MAXBD=BDNUM(N)
      MINBD=BDNUM(M)
      GC TO 7
    6 MAXBD=BDNUM(M)
      MINBD=BDNUM(N)
    7 DC 5 J=1,MAXND
      B=BDNUM(J)
      IF (B.EQ.MAXBD) BDNUM(J)=MINBD
      IF (B.GT.MAXBD) BDNUM(J)=BDNUM(J)-1
    5 CCNTINUE
      NUMBD=NUMBD-1
      GO TO 1
C
C        IF BOTH NODES ON THE ARC ARE UNASSIGNED, GO TO 4 WHERE A NEW
C        BUNDLE IS CREATED
C
    2 IF(BDNUM(N).EQ.0) GO TO 4
C
C        ASSIGN THE ORIGIN NODE OF THE ARC TO THE BUNDLE CCNTAINING THE
C        TERMINAL NODE
C
      BDNUM(M)=BDNUM(N)
      GC TC 1
C
C        ASSIGN THE TERMINAL NODE OF THE ARC TO THE BUNDLE CONTAINING
C        THE ORIGIN NODE OF THE ARC
C
    3 BDNUM(N)=BDNUM(M)
      GC TO 1
C
C        CREATE A NEW BUNDLE
C
    4 NUMBD=NUMBD+1
      BDNUM(M)=NUMBD
      BDNUM(N)=NUMBD
    1 CCNTINUE
515   CONTINUE
      BCNUM(SINK)=0
C
C        IF WE ONLY HAVE 1 BUNDLE FROM THE SUBNETWORK, WE ARE FINISHED
C
      IF (NUMBD.EQ.1) GO TO 219
C
C        ZEROIZE CHECK ARRAY
C
      DC 290 I=1,ARCS
290   CHECK(I)=0
      L=0
C
C        THE NODES ARE IN BUNDLES.  PUT THE ASSOCIATED ARCS INTO
```

```
C         APPROPRIATE PARALLEL SUBNETWORKS
C
      DO 33 I=1,NUMBD
      SUMARC=0
      NSUB=NSUB+1
      DO 34 K=1,ARCS
      M=SUBNET(K,TSUBN)
      N=S(M)
C
C         SOURCE AND SINK HAVE BUNDLE NUMBER 0
C
      IF (N.EQ.SOURCE) N=T(M)
      IF (BDNUM(N).EQ.I) GO TO 239
      IF (N.EQ.SINK) GO TO 229
      GO TO 34
C
C         SPECIAL CASE:  BUNDLE HAS ONLY 2 NODES: SOURCE, SINK.
C         PUT ALL ARCS THAT ARE PARALLEL SUBNETWORKS BY THEMSELVES INTO
C         THE CHECK ARRAY
C
229   DO 291 J=1,K
      W=CHECK(J)
      IF (W.EQ.0) GO TO 292
      IF (M.EQ.W) GO TO 34
291   CONTINUE
292   CHECK(J)=M
      GO TO 34
C
C         THIS ARC IS IN THE BUNDLE I, HENCE IT IS IN THE ITH NEW
C         SUBNETWORK
C
239   SUMARC=SUMARC+1
      SUBNET(SUMARC,NSUB)=M
34    CONTINUE
C
C         CREATE NEW SUBNETWORKS
C
C
C         IF THIS BUNDLE HAS NO NODES, PUT AN ARC SUBNETWORK INTO
C         SUBNET (1,NSUB)
C
      IF (SUMARC.EQ.0) GO TO 333
343   NARCSS(NSUB)=SUMARC
      SOURC(NSUB)=SOURCE
      SINKS(NSUB)=SINK
      GO TO 33
C
C         STORE THE SUBNETWORKS THAT HAVE ONLY SOURCE AND SINK NODES
C
333   SUMARC=1
      L=L+1
      SUBNET(1,NSUB)=CHECK(L)
      GO TO 343
33    CONTINUE
219   TNSUB=NUMBD
      RETURN
      END
      SUBROUTINE CUT (TSUBN,LNODEN)
C
C         CUT IDENTIFIES CUT NODES EXCLUDING THE DESIGNATED SOURCE AND
C         SINK
```

```
C
C        CUT ALSO IDENTIFIES THE CUT GROUPS; THAT IS, THE SUBNETWORKS
C        WHICH ARE IN SERIES AND CONNECTED BY THE CUT NODES
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      INTEGER*2 TARC(1000),ORIGIN(1000),POST(1000),RCUT(100),PPOST(1000)
C
C        FIND THE CUT NODES
C        NCUT IS THE NUMBER OF CUT NODES FOUND THUS FAR
C
      NCUT=0
      ARCS=NARCSS(TSUBN)
      SOURCE=SOURC(TSUBN)
      SINK=SINKS(TSUBN)
C
C        THE DO LOOP DOWN TO STATEMENT NUMBER 1 DETERMINES THE CUT NODES
C
      DO 1  K=1,LNODEN
C
C        CHECK TO SEE IF NODE K IS ACTUALLY IN THE SUBNETWORK
C
      DO 20 J=1,ARCS
      Z=SUBNET(J,TSUBN)
      IF(S(Z).EQ.K) GO TO 21
      IF (T(Z).EQ.K) GO TO 21
20    CONTINUE
C
C        NODE K IS NOT IN THIS SUBNETWORK
C
      GO TO 1
21    CONTINUE
C
C        NODE K IS IN THIS SUBNETWORK
C
      IF(K.EQ.SOURCE) GO TO 1
      IF(K.EQ.SINK) GO TO 1
      NTARC=0
      NPPOST=1
      DO 2 J=1,ARCS
      Z=SUBNET(J,TSUBN)
      IF (S(Z).EQ.K) GO TO 2
      IF (T(Z).EQ.K) GO TO 2
      NTARC=NTARC+1
      TARC(NTARC)=SUBNET(J,TSUBN)
2     CONTINUE
C
C        TARC IS THE SUBNETWORK WITHOUT THE ARCS INVOLVING NODE K
C        IF TARC CONTAINS A PATH FROM THE SOURCE TO THE SINK, THEN NODE
C        K IS NOT A CUT NODE
C        OTHERWISE, K IS A CUT NODE
C
      ORIGIN(1)=SOURCE
      PPOST(1)=SOURCE
      NORIG=1
11    CONTINUE
      NPOST=0
C
C        IF THERE ARE NO ARCS IN THE TARC ARRAY, K IS A CUT NODE
```

```
C
      IF (NTARC.EQ.0) GO TC 44
C
C        FIND ALL NODES WHICH COME AFTER AN ORIGIN:  PUT THEM IN POST
C        AND PPOST IF THEY ARE NOT PAST POSTS
C
      DO 4 I=1,NORIG
      Y=ORIGIN(I)
      DO 5 J=1,NTARC
      Z=TARC(J)
      U=S(Z)
      V=T(Z)
      IF (U.NE.Y) GO TO 5
C
C        IF WE'VE REACHED THE SINK, NODE K IS NOT A CUT NODE
C
C
      IF (V.EQ.SINK) GO TO 1
C
C        IF V IS A PAST POST, LET'S IGNORE IT
C
      DO 49 L=1,NPPOST
      IF (V.EQ.PPOST(L)) GO TO 5
 49   CONTINUE
      NPOST=NPOST+1
      POST(NPOST)=V
      NPPOST=NPPOST+1
      PPOST(NPPOST)=V
      GO TC 5
 5    CONTINUE
 4    CONTINUE
C
C        IF THERE ARE NOW NO POSTS, NODE K IS A CUT NODE
C
 44   IF(NPOST.NE.0) GO TO 13
      NCUT=NCUT+1
      RCUT(NCUT)=K
      GO TO 1
 13   NORIG=NPOST
      DO 14 L=1,NORIG
C
C        THESE ARE NOW OUR NEW ORIGINS
C
 14   ORIGIN(L)=POST(L)
C
C        CHECK THE NEW ORIGINS FOR THEIR POSTS
C
      GO TO 11
 1    CONTINUE
      IF (NCUT.EQ.0) GO TO 32
      NSUB=NSUB+1
      SOURC(NSUB)=SOURCE
 33   ORIGIN(1)=SOURCE
C
C        NOW WE NEED TO FIND THE COMPONENTS OF THE SERIES SUBNETWORKS
C        THAT ARE SEPARATED BY THE CUT NODES
C
 39   SUMARC=0
      NORIG=1
      NPPOST=1
 23   NPOST=0
```

```
      PPOST(1)=ORIGIN(1)
      DO 24 I=1,NORIG
      Y=ORIGIN(I)
      DO 25 J=1,ARCS
      Z=SUBNET(J,TSUBN)
C
C        ALL ARCS BEGINNING AT THIS ORIGIN GO INTO THE NEW SUBNETWORK
C
      IF (S(Z).NE.Y) GO TO 25
      SUMARC=SUMARC+1
      SUBNET (SUMARC,NSUB)=Z
C
C        T(Z) WILL BE A POST IF IT ISN'T A REPEAT OF A CURRENT
C        ORIGIN OR A CUT NODE
C
C        CHECK TO SEE IF IT IS A REPEAT
C
      X=T(Z)
      DO 30 K=1,NORIG
C
C        IF T(Z) IS A REPEAT OF A CURRENT ORIGIN, LET'S IGNORE IT
C
      IF(X.EQ.ORIGIN(K)) GO TO 25
30    CONTINUE
C
C        CHECK TO SEE IF T(Z) IS A CUT NODE.  IF IT IS, DON'T PUT IT
C        IN THE POST OR PPOST ARRAYS
C
      DO 52 W=1,NCUT
      D=RCUT(W)
      IF (X.NE.D) GO TO 52
C
C        THIS CUT NODE IS THE SINK OF THE SUBNETWORK UNDER
C        CONSIDERATION AND THE SOURCE OF THE NEXT SUBNETWORK TO BE
C        CONSIDERED.
C
      SINKS(NSUB)=D
      SOURC(NSUB+1)=D
      GO TO 25
52    CONTINUE
C
C        IF T(Z) IS A REPEAT OF A PAST POST, LET'S IGNORE IT
C
      DO 7 K=1,NPPOST
      IF (X.EQ.PPOST(K)) GO TO 25
7     CONTINUE
      NPOST=NPOST+1
      POST(NPOST)=X
      NPPOST=NPPOST+1
      PPOST(NPPOST)=X
25    CONTINUE
24    CONTINUE
C
C        IF WE HAVE NO POSTS LEFT, WE HAVE FOUND ALL OF THIS SUBNETWORK
C
      IF (NPOST.EQ.0) GO TO 34
      NORIG=NPOST
      DO 28 L=1,NORIG
28    ORIGIN(L)=POST(L)
      GO TO 23
34    NARCSS(NSUB)=SUMARC
```

```
      NSUB=NSUB+1
      X=SOURC(NSUB)
C
C        IF THE SOURCE OF NSUB IS NOT A CUT NODE, WE NEED TO ADJUST NSUB
C        AND GO BACK TO THE MAIN PROGRAM FOR THE NEXT STAGE OF THE
C        BREAKUP
C
      IF (X.EQ.0) GO TO 31
      ORIGIN(1)=X
      GO TO 39
31    NSUB=NSUB-1
      SINKS(NSUB)=SINK
32    TNSUB=NCUT+1
      RETURN
      END
      SUBROUTINE PUNSYN
C        THIS SUBROUTINE PUNCHES THE CARDS NEEDED FOR SYNTHESIS
C        CARD(I,1) = THE INSTRUCTION NUMBER,I
C        CARD(I,2) = THE DECOMPOSABLE SUBNETWORK
C        CARD(I,3) = ISORP;
C             ISORP=0 IMPLIES A SERIES DECOMPOSITION
C             ISORP=1 IMPLIES A PARALLEL DECOMPOSITION
C        CARD(I,4...) = THE SUBNETWORKS RESULTING FROM THE
C                       DECOMPOSITION OF SUBNETWORK I
      IMPLICIT INTEGER*2 (A-Z)
      COMMON /NEW/ CARD,INSTN
      INTEGER CARD(100,27),WHYNOT
      INTEGER                                 F5/13/
      WHYNOT = INSTN
      WRITE(F5) WHYNOT
      WRITE(F5) CARD
      ENDFILE F5
      RETURN
      END
      SUBROUTINE PRESUB(TSUBN)
C
C        PREPARES THE SUBNETWORK FOR SUBNETWORK ANALYSIS
C             INCLUDES RENUMBERING THE NODES 1,2,.... CONSECUTIVELY.
C
      IMPLICIT INTEGER*2 (A-Z)
      COMMON SUBNET,S,T,SOURC,SINKS,NARCSS,NSUB,TNSUB
      INTEGER*2 S(1000),T(1000),SUBNET(1000,100),SOURC(100),SINKS(100),
     *NARCSS(100)
      INTEGER*2 HEAD(1000),TAIL(1000),NODE(1000)
      COMMON/PREP/ NUMSUB
      INTEGER FILE3(1503),NET(3,500),N3
      INTEGER                       F3/11/
      EQUIVALENCE (FILE3(4),NET(1))
      NUMSUB=NUMSUB+1
C
C        TSUBN=THE NUMBER OF THE MINIMUM SUBNETWORK
C
C
C        RENUMBER THE NODES IN THE SUBNETWORK
C
      N=NARCSS(TSUBN)
      DO 1 I=1,N
      J=SUBNET(I,TSUBN)
      NODE(2*I-1) = S(J)
    1 NODE(2*I)   = T(J)
C
```

```
C-----SORT THE NODES
C
  100 LIMIT = 2*N
      IPASS = 2*N-1
      IFLG = 1
      DO 2 J=1,IPASS
      LIMIT = LIMIT-1
      IF(IFLG.EQ.0) GO TO 3
      IFLG = 0
      DO 4 K=1,LIMIT
      IF (NODE(K).LE.NODE(K+1)) GO TO 4
      KEEP = NODE(K+1)
      NODE(K+1) = NODE(K)
      NODE(K) = KEEP
      IFLG = 1
    4 CONTINUE
    2 CONTINUE
C
C-----DISCARD ANY DUPLICATIONS IN THE SORTED LIST
C
    3 KOUNT=1
      DO 5 I=1,IPASS
      IF (NODE(I).EQ.NODE(I+1)) GO TO 5
      KOUNT = KOUNT+1
      NODE(KOUNT) = NODE(I+1)
    5 CONTINUE
C
C-----CHANGE INPUT-HEAD,TAIL
C
      DO 8 L=1,N
      I=SUBNET(L,TSUBN)
      DO 9 J=1,KOUNT
      IF (    T(I).NE.NODE(J)) GO TO 9
      HEAD(L)=J
      GO TO 8
    9 CONTINUE
    8 CONTINUE
      DO 11 L=1,N
      I=SUBNET(L,TSUBN)
      DO 12 J=1,K
      IF (    S(I).NE.NODE(J)) GO TO 12
      TAIL(L)=J
      GO TO 11
   12 CONTINUE
   11 CONTINUE
      FILE3(1) = N
      FILE3(2) = TSUBN
      FILE3(3) = KOUNT
      DO 10 J=1,N
      NET(1,J) = HEAD(J)
      NET(2,J) = TAIL(J)
10    NET(3,J) = SUBNET(J,TSUBN)
C
C         OUTPUT
C
      WRITE(F3) FILE3(1)
      N3 = 2 + 3*N
      CALL WRT(F3,FILE3(2),N3)
      WRITE(6,200)
200   FORMAT(1H0,/16X,'THE FOLLOWING REPRESENTATION OF THE SUBNETWORK AB
     *OVE WILL BE USED BY THE SUBNETWORK ANALYSIS PROGRAM:')
```

```
      WRITE(6,30)
30    FORMAT(1H0,19X,'ACTIVITY',4X,'TAIL',6X,'HEAD')
      WRITE(6,31)                    (I,TAIL(I),HEAD(I),I=1,N)
31    FORMAT(20X,I5,5X,I5,5X,I5)
      WRITE(6,7005)
7005  FORMAT(1H1)
      RETURN
      END
      SUBROUTINE WRT(J,A,N)
      DIMENSION A(N)
      WRITE(J) A
      RETURN
      END
      SUBROUTINE RD(J,A,N)
      DIMENSION A(N)
      READ(J) A
      RETURN
      END
```

7. SUBNET

```
C         SUBNETWORK ANALYSIS PROGRAM
C
C         THIS PROGRAM DETERMINES THE DURATION DISTRIBUTION FOR EACH
C         SUBNETWORK IDENTIFIED BY THE DECOMPOSITION PROGRAM.
C
C         FOR THE SAKE OF  IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C         MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                    PROJECT NETWORK
C         LMAX = THE MAXIMUM NUMBER OF ACTIVITIES ON A CRITICAL PATH
C                    LMAX IS LESS THAN OR EQUAL TO MSMAX
C         CMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN A CLUSTER
C                    CMAX IS LESS THAN OR EQUAL TO MSMAX
C         MSMAX = THE MAXIMUM NUMBER OF ACTIVITIES ALLOWED IN ANY ONE
C                    SUBNETWORK
C         NSUBMAX = THE MAXIMUM NUMBER OF SUBNETWORKS ALLOWED IN THE
C                    DECOMPOSITION PROCESS
C         IEDFMAX = THE MAXIMUM NUMBER OF SUBDIVISIONS ALLOWED IN THE
C                    APPROXIMATE CDF FOR EACH SUBNETWORK
C         SDIM = 12 * (IEDFMAX-1) + 2
C         CURRENTLY, MMAX=1000; LMAX=50; CMAX=100; MSMAX=500; NSUBMAX=100
C                    IEDFMAX=20; SDIM=230
C
C      DIMENSION CTIME(MSMAX),DTIME(MSMAX),AORD(MSMAX),TAIL(MSMAX),
C     *          HEAD(MSMAX),XNODE(MSMAX),IBB(LMAX),KB(LMAX),ICRITP(LMAX)
C     *          COT(MSMAX),SIGMA(MSMAX),NINCL(LMAX),INCLUS(LMAX,CMAX),
C     *          FD(IEDFMAX),ULEFD(IEDFMAX),LLEFD(IEDFMAX),FHI(MSMAX),
C     *          FLO(MSMAX),PP(MSMAX),PC(MSMAX),FLHAT(IEDFMAX),
C     *          FUHAT(IEDFMAX),FDLHAT(3,IEDFMAX),FDUHAT(3,IEDFMAX),
C     *          FHAT(IEDFMAX),BRHS1(SDIM),INBASE(SDIM),XB1(SDIM),
C     *          Y1(SDIM),B1INV(SDIM,SDIM),FILE3(3*MSMAX+2),NET(3,MSMAX),
C     *          LUP(3,MMAX)
C
C         IN SUBROUTINE ORDER THE ARRAYS ARE:
C      DIMENSION ND(MSMAX),NDD(MSMAX)
C
C         IN SUBROUTINE CLUSTR THE ARRAYS ARE:
C      DIMENSION LEFT(MSMAX),LEFTO(MSMAX),NONCP(MSMAX),NINAG(MSMAX),
C     *          NCLINC(LMAX),ASSGRP(LMAX,LMAX),CLINCL(LMAX,LMAX),
C     *          EGRP(LMAX)
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       IMPLICIT REAL*8 (A-H,O-Z)
       COMMON /BLKA/CTIME,DTIME,M
       DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       COMMON /BLAA/XNODE,TAIL,HEAD,AORD,NMM
       INTEGER AORD(500),TAIL(500),HEAD(500)
       DIMENSION XNODE(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C      COMMON /BLKB/IBB,KB,KKB,KKK
C      DIMENSION IBB(50),KB(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       COMMON /BLKC/ICRITP,KCPB
       DIMENSION ICRITP(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       COMMON /BLKD/SIGMA,COT
       DIMENSION COT(500),SIGMA(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       COMMON /BLKE/INCLUS,NINCL,NCLUS
       DIMENSION NINCL(50),INCLUS(50,100)
```

```
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),PQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKJ/THET,LAMBD
      REAL*8 LAMBD
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKL/FLHAT,FUHAT
      DIMENSION FLHAT(20),FUHAT(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKY/FDUHAT,FDLHAT,FHAT
      DIMENSION FDLHAT(3,20),FDUHAT(3,20),FHAT(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKZ/THETA
      REAL*4 THETA(3)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C     COMMON /BLK1/B1INV,OMEGA,MM,KK,MMKK,MMKK2,N,MP1,JFIRST
C     COMMON /BLK2/Y1
C     DIMENSION BRHS1(230),INBASE(230),XB1(230),Y1(230),B1INV(230,230)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      INTEGER THELAM
      INTEGER FILE3(1502)
      INTEGER                    F3/11/,F4/12/,F5/13/
      REAL*8 LUP(3,1000)
      DIMENSION NET(3,500)
      DIMENSION NIMPT(3)
      DIMENSION CDF(12)
      REAL*4 LAMBDA(3)
      REAL*4 RTHETA(3),RLAMBD(3)
      REAL*8 M1,M2,M3
      EQUIVALENCE (FILE3(3),NET(1))
      EQUIVALENCE (FILE3(6),RTHETA(1)),(FILE3(9),RLAMBD(1))
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C
      WRITE(6,90001)
90001 FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FROM THE SUBN
     *ETWORK ANALYSIS PROGRAM: SUBNET'/1H0,132('*')/1X,132('*')/////
     *' EACH SUBNETWORK',1H','S APPROXIMATE DURATION DISTRIBUTION IS DET
     *ERMINED.')
C*********************************************************************
C
C         READ IN INFORMATION PERTAINING TO ALL SUBNETWORKS
C
C*********************************************************************
      REWIND F3
      REWIND F5
      READ(F5)
      READ(F5)
      READ(F5)
      REWIND F4
      READ(F4)
```

```
      READ(F4) NACT
C***********************************************************************
C
C         DETERMINE L,U, AND P FOR EACH ACTIVITY IN THE SIMPLIFIED
C         PROJECT NETWORK
C
C***********************************************************************
      DO 10 L=1,NACT
      READ(F4)CDF
      M1=0.D0
      M2=0.D0
      M3=0.D0
      DO 20 K=2,11
      M1=M1+CDF(K)
      M2=M2+CDF(K)**2
20    M3=M3+CDF(K)**3
      M1=M1*.1D0
      M2=M2*.1D0
      M3=M3*.1D0
      T3=M1*M1-M2
      IF(DABS(T3).GE.1.D-10)GO TO 21
      U=CDF(2)
      B=U
      P=1.D0
      GO TO 22
21    T1=(M3-M1*M2)**2
      T2=4.D0*T3*(M2*M2-M1*M3)
      T3=2.D0*T3
      U=-M3+M1*M2-DSQRT(T1-T2)
      U=U/T3
      B=(M2-M1*U)/(M1-U)
      P=(M1-U)/(B-U)
22    LUP(1,L)=B
      LUP(2,L)=U
      LUP(3,L)=P
10    CONTINUE
      CALL RD(F3,FILE3,11)
      IEDF=FILE3(1)
      NMAX=FILE3(2)
      IPCOL=FILE3(3)
      SAMSIZ=FILE3(4)
      THELAM=FILE3(5)
      DO 2 I=1,3
      THETA(I)=RTHETA(I)
2     LAMBDA(I)=RLAMBD(I)
1     CONTINUE
C***********************************************************************
C
C         ANALYZE EACH SUBNETWORK SEPARATELY
C
C
C         READ IN THE DESCRIPTION OF THE SUBNETWORK
C
C***********************************************************************
      READ(F3,END=999) M
      N3 = 2 + 3*M
      CALL RD(F3,FILE3(1),N3)
      NSUB = FILE3(1)
      NWW  = FILE3(2)
      WRITE(6,2600) NSUB
2600  FORMAT(1H1, 5X,'SUBNETWORK ',I5)
```

```
C        M = THE NUMBER OF ACTIVITIES IN THE SUBNETWORK
C        NMM = THE NUMBER OF NODES IN THE SUBNETWORK
      MP1=M+1
C
C        THE ACTIVITIES ARE DESCRIBED IN TERMS OF THEIR NODES
C        II=THE TAIL NODE. THE ORIGIN NODE
C        JJ=THE HEAD NODE. THE TERMINAL NODE
C        FLC = THE LOWER POINT
C        FHI = THE UPPER POINT
C        SIGMA  = (FHI - FLO)*DSQRT( P*(1-P) ) = STD. DEVIATION
C        PP = THE PROBABILITY OF THE LOWER POINT
C
      DO 610 I=1,M
      HEAD(I) = NET(1,I)
      TAIL(I) = NET(2,I)
      K       = NET(3,I)
      FLC(I)  = LUP(1,K)
      FHI(I)  = LUP(2,K)
      PP(I)   = LUP(3,K)
      PQ(I)=1.D0-PP(I)
      SIGMA(I)=(FHI(I)-FLO(I))*DSQRT( PP(I)*PQ(I))
      CCT(I)=PP(I)*FLO(I)+PQ(I)*FHI(I)
      CTIME(I) = COT(I)
610   CONTINUE
C        COT = THE ORIGINAL RIGHT-HAND SIDES. I.E. THE MEANS
C              OF FLO AND FHI
C        CTIME = THE CURRENT RIGHT-HAND SIDES
      WRITE(6,2700)
2700  FORMAT(1H0,10X,'INITIAL INPUT')
      WRITE(6,2701)
2701  FORMAT(1H0,10X,'ACTIVITY  ORIGIN    TERMINAL  LOWER POINT  UPPER P
     *OINT     MEAN     STANDARD DEVIATION   PROB. LOWER PT.')
      DO 2704 I=1,M
2704  WRITE(6,2702) I,TAIL(I),HEAD(I),FLO(I),FHI(I),CTIME(I),SIGMA(I),PP
     *(I)
2702  FORMAT(1H ,13X,I3,5X,I3,7X,I3,5X,F10.4,3X,F10.4,3X,F10.4,4X,F10.4,
     *14X,F7.5)
C***********************************************************************
C
C        IF THERE IS ONLY 1 ACTIVITY IN THE SUBNETWORK, LET  FHAT  BE
C        THAT ACTIVITY'S DURATION DISTRIBUTION
C
C***********************************************************************
      IF(M.GT.1)GO TO 4
      REWIND F4
      I=K+1
      DO 25 L=1,I
      READ(F4)
25    CONTINUE
      READ(F4)(  FD(I),I=1,12)
      JEDF=12
      FHAT(1)=0.D0
      FHAT(12)=1.D0
      M1=-.05D0
      DO 30 L=2,11
      M1=M1+.1D0
30    FHAT(L)=M1
      WRITE(6,700)NSUB
700   FORMAT(1H1,5X,'THE DURATION DISTRIBUTION FOR SUBNETWORK ',I2)
      WRITE(6,108)
      DO 701 I=1,JEDF
```

```
701   WRITE(6,103)FD(I),FHAT(I)
      GO TO 3
C***********************************************************************
C
C        INITIAL  DETERMINES THE CRITICAL PATH WHEN EACH
C        ACTIVITY'S DURATION IS ITS MEAN, LOWER BOUND, AND UPPER BOUND
C
C***********************************************************************
4     CALL INITAL
C***********************************************************************
C
C        IF M IS LESS THAN OR EQUAL TO NMAX, LET  FHAT  BE THAT
C        ACTIVITY'S DISCRETE DURATION DISTRIBUTION
C
C***********************************************************************
      IF(M.GT.NMAX)GO TO 5
      CALL FINDF
      JEDF=IEDF
      GO TO 3
C***********************************************************************
C
C        DETERMINE THETA AND LAMBDA
C
C        IF THELAM = 0 THE DEFAULT PAIRS (THETA,LAMBDA)=(1,1),(2,2),(3,2
C        WILL BE USED
C
C        IF THELAM IS NOT 0 THE USER-SPECIFIED (THETA,LAMBDA) PAIRS
C        WILL BE USED
C
C        NOTE: THE FOLLOWING RELATIONS MUST ALWAYS BE TRUE
C              THETA(1)<=THETA(2)<=THETA(3)
C              LAMBDA(1)<=LAMBDA(2)<=LAMBDA(3)
C
C***********************************************************************
5     DO 500 I=1,3
      IF(THELAM.NE.0)GO TO 222
      LAMBD=2.D0
      IF(I.EQ.1)LAMBD=1.D0
      THET=DFLOAT(I)
      THETA(I)=THET
      GO TO 201
222   THET=THETA(I)
      LAMBD=LAMBDA(I)
C***********************************************************************
C
C        SUBROUTINE CLUSTR  DETERMINES THE CLUSTER FOR GIVEN THETA AND
C        LAMBDA
C
C***********************************************************************
201   CALL CLUSTR
C
C        THE FOLLOWING DETERMINES WHETHER OR NOT THESE CLUSTERS HAVE
C        BEEN CONSIDERED BEFORE
C
      ISUM=0
      DO 200 II=1,NCLUS
200   ISUM=ISUM+NINCL(II)
      NIMPT(I)=ISUM
      IF(I.EQ.1)GO TO 202
      IF(ISUM.EQ.NIMPT(I-1))GO TO 203
202   CONTINUE
```

```
C*****************************************************************************
C
C        CALCULATE THE UPPER AND LOWER BOUNDS ON THE SUBNETWORK
C        DURATION DISTRIBUTION
C
C*****************************************************************************
C
C        ELIMINATE FROM THE CLUSTERS ANY ACTIVITIES HAVING CONSTANT
C        DURATION.  IF THESE ACTIVITIES ARE LEFT IN, THE TOTAL NUMBER
C        OF POSSIBLE ACTIVITY DURATION CONFIGURATIONS FOR THE CLUSTERS
C        IS ARTIFICIALLY INFLATED.
C
      DO 250 II=1.KCPB
      IL=NINCL(II)
      IF(IL.LE.1)GO TO 250
      III=1
251   IF(III.GT.IL)GO TO 254
      L=INCLUS(II.III)
      IF(SIGMA(L).NE.0.D0)GO TO 253
      IL=IL-1
      IF(III.GT.IL)GO TO 254
      DO 252 IIII=III.IL
252   INCLUS(II.IIII)=INCLUS(II.IIII+1)
253   III=III+1
      GO TO 251
254   NINCL(II)=IL
      IF(IL.EQ.0)NCLUS=NCLUS-1
250   CONTINUE
      IF(IPOOL.EQ.1) CALL UNION
      IF(IPOOL.EQ.0) CALL SINGLE
C*****************************************************************************
C
C        SAVE THE ESTIMATES CALCULATED BY  UNION  OR  SINGLE
C        FOR THE  I-TH THETA-LAMBDA PAIR
C
C*****************************************************************************
203   CONTINUE
      WRITE(6.100) NSUB.THET.LAMBD
100   FORMAT(1H1. 5X.'BOUNDS ON THE DURATION DISTRIBUTION FOR SUBNETWORK
     * '.       I4.' WHEN (THETA.LAMBDA) = ( '.E15.5.' . '.E15.5.' )')
      WRITE(6.101)
101   FORMAT(1H0.14X.'TIME'.16X.'LOWER BOUND'.9X.'UPPER BOUND'./)
      IFLG=0
      DO 300 II=1.IEDF
      FLTEMP=FLHAT(II)
      FUTEMP=FUHAT(II)
      WRITE(6.103)FD(II).FLTEMP.FUTEMP
103   FORMAT(1H .10X.3(E15.5.5X))
      IF(FLTEMP.NE.FUTEMP)IFLG=1
      FHAT(II)=FLTEMP
      FDLHAT(I.II)=FLTEMP
300   FDUHAT(I.II)=FUTEMP
      IF(IPOOL.EQ.1)WRITE(6.350)
      IF(IPOOL.EQ.0)WRITE(6.351)
350   FORMAT(1H0./11X.'THE BOUNDS WERE DETERMINED USING THE UNION-CLUSTE
     *R PROCEDURE.')
351   FORMAT(1H0./11X.'THE BOUNDS WERE DETERMINED USING THE MAXIMUM-CLUS
     *TER PROCEDURE.')
      IF(ICLGIB)352.353.353
352   WRITE(6.6645)
6645  FORMAT(1H0./11X.'ALL ACTIVITY DURATION CONFIGURATIONS WERE CONSIDE
```

```
     *REC.')
      GO TO 500
353   IF(IPOOL)357,357,358
358   WRITE(6,6046)SAMSIZ
6046  FORMAT(1H0,/11X,'ONLY ',I8,' OF THE POSSIBLE ACTIVITY DURATION CON
     *FIGURATIONS WERE CONSIDERED.')
      IF(ICLGIB.EQ.0)WRITE(6,105)
      IF(ICLGIB.EQ.1)WRITE(6,106)
      GO TO 500
357   WRITE(6,6047)SAMSIZ
6047  FORMAT(1H0,/11X,   'ONLY ',I8,' OF THE POSSIBLE ACTIVITY DURATION
     *CONFIGURATIONS FOR EACH CLUSTER WERE CONSIDERED.')
      IF(ICLGIB.EQ.0) WRITE(6,105)
      IF(ICLGIB.EQ.1) WRITE(6,106)
105   FORMAT(1H0,10X,'SYSTEMATIC SAMPLING WAS USED.')
106   FORMAT(1H0,10X,'SIMPLE RANDOM SAMPLING WAS USED.')
500   CONTINUE
C
C        IF THE BEST UPPER AND LOWER BOUNDS COMPUTED ABOVE ARE EQUAL,
C        THERE IS NO NEED TO PERFORM THE EXTRAPOLATION
C
C        IFLG = 0 IMPLIES THE BOUNDS ARE EQUAL
C        IFLG = 1 IMPLIES THE BOUNDS ARE NOT EQUAL
C
      IF(IFLG.EQ.1)GO TO 802
801   WRITE(6,803)NSUB
803   FORMAT(1H1,5X,'THE UPPER AND LOWER BOUNDS ON THE DISCRETE SUBNETWO
     *RK DURATION DISTRIBUTION ARE EQUAL.'//6X,'THUS AN ESTIMATE OF THE
     *DISCRETE DURATION DISTRIBUTION FOR SUBNETWORK ',I4,' IS')
      JEDF=IEDF
      GO TO 804
C*********************************************************************
C
C        SUBROUTINE SIMPLX  INTERPOLATES BETWEEN THE UPPER AND LOWER
C        BOUNDS DETERMINED ABOVE AND COMPUTES AN ESTIMATED COMPLETION
C        TIME DISTRIBUTION FOR THE SUBNETWORK BEING CONSIDERED
C
C*********************************************************************
802   CALL SIMPLX
9     JEDF=IEDF
      WRITE(6,107)NSUB
107   FORMAT(1H1,5X,'AN ESTIMATE OF THE DISCRETE DURATION DISTRIBUTION F
     *OR SUBNETWORK ',I2)
804   WRITE(6,108)
108   FORMAT(1H0,14X,'TIME',16X,'DISTRIBUTION'/)
      DO 9471 I=1,JEDF
9471  WRITE(6,103)FD(I),FHAT(I)
3     CONTINUE
      WRITE(F5)NSUB,JEDF,(FD(I),FHAT(I),I=1,JEDF)
      GO TO 1
999   WRITE(6,850)
850   FORMAT(1H1)
      ENDFILE F5
      STOP
      END
      SUBROUTINE RD(J,A,N)
      DIMENSION A(N)
      READ(J) A
      RETURN
      END
      SUBROUTINE INITAL
```

```
C
C         INITIAL  DETERMINES THE CRITICAL PATH WHEN EACH
C         ACTIVITY'S DURATION IS ITS MEAN, LOWER BOUND, AND UPPER BOUND
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKB/IBB,KB,KKB,KKK
      DIMENSION IBB(50),KB(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKC/ICRITP,KCPB
      DIMENSION ICRITP(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),PQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      CALL ORDER
      TOTAL=CPTIME(CPATHT)
      CALL FPATH
C
C          ICRITP(L)= THE L-TH ACTIVITY ON THE CRITICAL PATH
C           KCPB= THE NUMBER OF ACTIVITIES ON THE CRITICAL PATH
C
      KCPB=KKB
      DO 2802 I=1,KCPB
2802  ICRITP(I)=IBB(I)
      X=TOTAL
      WRITE(6,851) X
851   FORMAT(1H0,5X,'THE CRITICAL PATH TIME WHEN EACH ACTIVITY''S COMPLE
     *TION TIME IS SET EQUAL TO ITS MEAN IS = ',D15.5)
      WRITE(6,7606) KKK
 7606 FORMAT(1H0,10X,'THE ',I3,' NODES ON THE CRITICAL PATH ARE AS FOLLO
     *WS BEGINNING WITH THE TERMINAL NODE:')
      WRITE(6,7707) (KB(I),I=1,KKK)
 7707 FORMAT(15X,20(I3,','))
      WRITE(6,7710) KKB
7710  FORMAT(1H0,10X,'THE ',I3,' CRITICAL ACTIVITIES ARE AS FOLLOWS BEGI
     *NNING WITH THE TERMINAL ACTIVITY:')
      WRITE(6,7707) (IBB(I),I=1,KKB)
C
C         THE COMPLETION TIME FOR ALL ACTIVITIES IS SET TO THEIR LOWER
C         PERCENTILE.  THE RESULTING LONGEST PATH TIME IS A LOWER
C         BOUND ON THE SUBNETWORK DURATION TIME.
C
      DO 6402 I=1,M
 6402 CTIME(I) = FLO(I)
      CPLB=CPTIME(CPATHT)
      CALL FPATH
      WRITE(6,6405) CPLB
6405  FORMAT(1H0,5X,'A LOWER BOUND ON THE SUBNETWORK DURATION TIME IS =
     *',E15.5)
      WRITE(6,8606) KKK
8606  FORMAT(1H0,10X,'THE ',I3,' NODES ON THE LONGEST PATH ARE AS FOLLOW
```

```
     *S BEGINNING WITH THE TERMINAL NODE:')
      WRITE(6,7707) (KB(I),I=1,KKK)
      WRITE(6,8710) KKB
 8710 FORMAT(1H0,10X,'THE ',I3,' ACTIVITIES ON THE LONGEST PATH ARE AS F
     *OLLOWS BEGINNING WITH THE TERMINAL ACTIVITY:')
      WRITE(6,7707) (IBB(I),I=1,KKB)
C
C        THE COMPLETION TIME FOR ALL ACTIVITIES IS SET TO THEIR UPPER
C        PERCENTILE.  THE RESULTING LONGEST PATH TIME IS AN UPPER BOUND
C        ON THE SUBNETWORK DURATION TIME.
C
      DO 6406 I=1,M
 6406 CTIME(I) = FHI(I)
      CPUB=CPTIME(CPATHT)
      CALL FPATH
      WRITE(6,6409) CPUB
 6409 FORMAT(1H0,5X,'A UPPER BOUND ON THE SUBNETWORK DURATION TIME IS =
     *',E15.5)
      WRITE(6,8606) KKK
      WRITE(6,7707) (KB(I),I=1,KKK)
      WRITE(6,8710) KKB
      WRITE(6,7707) (IBB(I),I=1,KKB)
C
C        FD(I) = THE LOWER BOUND ON THE SUBNETWORK DURATION TIME
C                PLUS THE I-TH FRACTION OF THE DISTANCE TO THE UPPER
C                BOUND
C
C        FD  IS USED TO BUILD AN 'EMPIRICAL' DISTRIBUTION OF THE
C        SUBNETWORK DURATION TIMES
C
      C=CPUB-CPLB
      CC=.8D0*C
      LEDF=.6D0*IEDF
      CD=CC/LEDF
      DO 6412 I=1,LEDF
 6412 FD(I)=CPLB+I*CD
      CD=(C-CC)/(IEDF-LEDF)
      II=LEDF+1
      C=CPLB+CC
      DO 6512 I=II,IEDF
 6512 FD(I)=C+(I-LEDF)*CD
      RETURN
      END
      SUBROUTINE ORDER
C        THIS SUBROUTINE DETERMINES THE ORDER IN WHICH TO CONSIDER
C        THE ACTIVITIES FOR THE CALCULATION OF THE LONGEST  PATH TIME
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLAA/XNODE,TAIL,HEAD,AORD,NMM
      INTEGER AORD(500),TAIL(500),HEAD(500)
      DIMENSION XNODE(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DIMENSION ND(500),NDD(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      N=NMM
      NDD(1)=1
      DO 5 I=2,N
    5 NDD(I)=0
```

```
      DO 6 I=1,M
6     AORD(I)=0
      K=0
      MP=M+1
      DO 1 II=1,MP
      DO 20 I=1,N
20    NO(I)=NOD(I)
      III=0
      IP=II+1
      DO 2 J=1,M
      IF(NO(TAIL(J)).NE.II) GO TO 2
      NOD(HEAD(J))=IP
      III=1
      IF(K.EQ.0) GO TO 14
      DO 10 L=1,K
      IF(AORD(L).EQ.J) GO TO 11
10    CONTINUE
14    K=K+1
      GO TO 13
   11 IF(L.EQ.K) GO TO 2
      KM=K-1
      DO 12 LL=L,KM
12    AORD(LL)=AORD(LL+1)
13    AORD(K)=J
2     CONTINUE
      IF(III.EQ.0) GO TO 3
1     CONTINUE
3     CONTINUE
      WRITE(6,70)
70    FORMAT(1H1,5X,'THE ORDER IN WHICH TO CONSIDER THE ACTIVITIES TO DE
     *TERMINE THE LONGEST PATH TIME:')
      WRITE(6,71)
71    FORMAT(1H0,10X,'ORDER  ACTIVITY')
      DO 50 I=1,M
50    WRITE(6,51) I,AORD(I)
51    FORMAT(1H ,10X,I5,5X,I5)
      RETURN
      END
      FUNCTION CPTIME(CPATHT)
C        DETERMINE THE CRITICAL PATH TIME: CPTIME
C        XNODE(I) = EARLIEST TIME THAT AN ACTIVITY BEGINNING AT NODE I
C                 CAN COMMENCE
C
      IMPLICIT REAL*8 (A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLAA/XNODE,TAIL,HEAD,AORD,NMM
      INTEGER AORD(500),TAIL(500),HEAD(500)
      DIMENSION XNODE(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DO 1 I=1,NMM
1     XNODE(I)=0.D0
      DO 2 II=1,M
      I=AORD(II)
2     XNODE(HEAD(I))=DMAX1(XNODE(TAIL(I))+CTIME(I),XNODE(HEAD(I)))
      I=HEAD(AORD(M))
      CPTIME=XNODE(I)
      RETURN
      END
```

```
      FUNCTION CLTIME(CPATHT)
C        DETERMINE THE CRITICAL PATH TIME: CLTIME
C        XNODE(I) = EARLIEST TIME THAT AN ACTIVITY BEGINNING AT NODE I
C                 CAN COMMENCE
C
      IMPLICIT REAL*8 (A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLAA/XNODE,TAIL,HEAD,AORD,NMM
      INTEGER AORD(500),TAIL(500),HEAD(500)
      DIMENSION XNODE(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DO 1 I=1,NMM
1     XNODE(I)=0.D0
      DO 2 II=1,M
      I=AORD(II)
2     XNODE(HEAD(I))=DMAX1(XNODE(TAIL(I))+CTIME(I),XNODE(HEAD(I)))
      I=HEAD(AORD(M))
      CLTIME=XNODE(I)
      RETURN
      END
      SUBROUTINE FPATH
C
C        SUBROUTINE FPATH  FINDS THE CRITICAL PATH
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLAA/XNODE,TAIL,HEAD,AORD,NMM
      INTEGER AORD(500),TAIL(500),HEAD(500)
      DIMENSION XNODE(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKB/IBB,KB,KKB,KKK
      DIMENSION IBB(50),KB(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C
C        KKK= THE NUMBER OF NODES ON THE CRITICAL PATH
C        KB(L)= THE L-TH NODE IN THE CRITICAL PATH, COUNTING BACKWARDS
C               FROM THE TERMINAL NODE
C        KKB= THE NUMBER OF ACTIVITIES ON THE CRITICAL PATH
C        IBB(L)= THE L-TH ACTIVITY ON THE CRITICAL PATH, COUNTING
C                BACKWARDS FROM THE TERMINAL NODE
C
C
C        THE FOLLOWING STATEMENTS DETERMINE THE NODES AND ACTIVITIES ON
C        THE CRITICAL PATH
C
      KKK=1
      KB(1)=NMM
83001 IK=KB(KKK)
      SMIN=999999.
      ISMIN=0
      DO 83000 I=1,M
      IF(HEAD(I).NE.IK) GO TO 83000
      SLACK=XNODE(HEAD(I))-XNODE(TAIL(I))-CTIME(I)
      IF(SLACK.GE.SMIN) GO TO 83000
      SMIN=SLACK
```

```
      ISMIN=I
83000 CONTINUE
83003 IBB(KKK)=ISMIN
      KKK=KKK+1
      KB(KKK)=TAIL(ISMIN)
      IF(TAIL(ISMIN).GT.1) GO TO 83001
      KKB=KKK-1
      RETURN
      END
      SUBROUTINE FINDF
C
C         SUBROUTINE FINDF EVALUATES THE SUBNETWORK'S DISCRETE DURATION
C         DISTRIBUTION BY EXPLICITLY EVALUATING ALL OF THE SUBNETWORK'S
C         ACTIVITY DURATION CONFIGURATIONS
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKY/FDUHAT,FDLHAT,FHAT
      DIMENSION FDLHAT(3,20),FDUHAT(3,20),FHAT(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IR=1
C***********************************************************************
C
C         FORM A CLUSTER CONTAINING ALL OF THE ACTIVITIES IN THE
C         SUBNETWORK
C
C***********************************************************************
      NINCL(1)=M
      DO 1501 I=1,M
1501  INCLUS(1,I)=I
      DO 7101 I=1,IEDF
7101  LLEFD(I)=0.D0
      IB=2**M
C***********************************************************************
C
      CALL EXACTF
C
C***********************************************************************
      FHAT(1)=LLEFD(1)/SUM
      DO 6900 I=2,IEDF
      II=I-1
      LLEFD(I)=LLEFD(II)+LLEFD(I)
      FHAT(I)=LLEFD(I)/SUM
6900  CONTINUE
      WRITE (6,6362)
```

```
6362  FORMAT (1H1)
      WRITE(6,100) NSUB
100   FORMAT(1H1, 5X,'THE DISCRETE DURATION DISTRIBUTION FOR SUBNETWORK
     *',I2)
      WRITE(6,101)
101   FORMAT(1H0,14X,'TIME',16X,'DISTRIBUTION'/)
      DO 102 I=1,IEDF
102   WRITE(6,103) FD(I),FHAT(I)
103   FORMAT(1H ,10X,2(E15.5,5X))
      RETURN
      END
      SUBROUTINE EXACTF
C
C        SUBROUTINE EXACTF DETERMINES THE EMPIRICAL DISTRIBUTION
C        FUNCTION FOR THE SUBNETWORK DURATIONS CORRESPONDING TO ALL
C        POSSIBLE ACTIVITY DURATION CONFIGURATIONS
C
      IMPLICIT REAL*8(A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IJ=0
      NIB=0
      KJ=NINCL(IR)
      SUM=0.D0
2001  NIB=NIB+1
      IF(NIB.GT.IB) GO TO 2000
      CALL CONVRT(NIB,IJ,KJ,SPROB)
      XL=CLTIME(CPATHT)
      X=XL-1.D-10
      I=0
6420  I=I+1
      IF(X.GT.FD(I)) GO TO 6420
      LLEFD(I) = LLEFD(I) + SPROB
      SUM=SUM+SPROB
      GO TO 2001
2000  CONTINUE
      NIB=NIB-1
      RETURN
      END
      SUBROUTINE CLUSTR
C
C        SUBROUTINE CLUSTER DETERMINES THE CLUSTER FOR GIVEN THETA AND
C        LAMBDA
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKB/IBB,KB,KKB,KKK
      DIMENSION IBB(50),KB(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

```
      COMMON /BLKC/ICRITP,KCPB
      DIMENSION ICRITP(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKD/SIGMA,COT
      DIMENSION COT(500),SIGMA(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKJ/THETA,LAMBDA
      REAL*8 LAMBDA,THETA
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DIMENSION LEFT(500),LEFTO(500),NONCP(500),NINAG(500)
      INTEGER ASSGRP(50,50),CLINCL(50,50),EGRP(50),NCLINC(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C
C         THE ASSOCIATE GROUPS ARE NOW FORMED
C
      WRITE(6,3165)LAMBDA
 3165 FORMAT(1H1,5X,'THE ASSOCIATES ARE NOW IDENTIFIED FOR LAMBDA = ',
     *E15.5)
      IIIII=1
      DO 2825 I=1,M
 2825 CTIME(I)=COT(I)
      IWWWQ=ICRITP(1)
 2801 CONTINUE
22825 CTIME(IWWWQ)=COT(IWWWQ)
      IWWWQ=ICRITP(IIIII)
      TEX=COT(IWWWQ)-LAMBDA*SIGMA(IWWWQ)
      IF(TEX.LT.0.D0)TEX=0.D0
      CTIME(IWWWQ)=TEX
      DUMMY=CPTIME(CPATHT)
      CALL FPATH
C
C         DETERMINE ASSOCIATE GROUP
C
 2910 NINAG(IIIII)=0
      DO 2911 K=1,KKB
      KK=1
 2913 IF(IBB(K).EQ.ICRITP(KK)) GO TO 2911
      IF(KK.GE.KCPB) GO TO 2912
      KK=KK+1
      GO TO 2913
 2912 NINAG(IIIII)=NINAG(IIIII)+1
      ASSGRP(IIIII,NINAG(IIIII))=IBB(K)
 2911 CONTINUE
      WRITE(6,2915) IIIII,ICRITP(IIIII),NINAG(IIIII)
 2915 FORMAT(1H0,10X,'THE NUMBER OF ASSOCIATES ASSOCIATED WITH THE ',I3,
     *'-TH CRITICAL PATH ACTIVITY, I.E. ACTIVITY ',I3,', IS = ',I3)
      IDUCK=NINAG(IIIII)
      IF(IDUCK.EQ.0) GO TO 2810
      WRITE(6,2916) (ASSGRP(IIIII,I),I=1,IDUCK)
 2916 FORMAT(1H0,15X,'THE ACTIVITIES IN THE ASSOCIATE GROUP ARE AS FOLLO
     *WS',/,15X,50(I3,','))
 2810 IIIII=IIIII+1
      IF(IIIII.LE.KCPB) GO TO 2801
C
C         DETERMINE THE CLUSTERS
C
C               THE CLUSTERS ARE POOLED TOWARD THE TERMINAL NODE
C               NCLUS = THE NUMBER OF NON-EMPTY CLUSTERS
```

```
C             NINCL(I) = THE NUMBER OF ACTIVITIES IN THE I-TH CLUSTER
C             INCLUS(I,J) = THE J-TH ACTIVITY IN THE I-TH CLUSTER
C             NCLINC(I) = THE NUMBER OF CLUSTERS COMPRISING THE I-TH
C                  CLUSTER AFTER POOLING
C             CLINCL(I,J) = THE J-TH CLUSTER WHICH HAS BEEN POOLED INTO
C                  THE I-TH CLUSTER
C
C             NCLINC AND CLINCL HELP KEEP TRACK OF WHICH CLUSTER THE
C             CRITICAL PATH ACTIVITIES ARE IN
C
C
C        BELOW FORMS CLUSTERS BY PUTTING EACH CRITICAL PATH ACTIVITY IN
C        SEPARATE CLUSTER AND THEN ADDING EACH CRITICAL PATH ACTIVITY'S
C        ASSOCIATES TO ITS CLUSTER
C
      NCLUS=KCPB
      DO 3020 I=1,KCPB
      NCLINC(I)=1
      CLINCL(I,1)=I
      NINCL(I) =NINAG(I)+1
      INCLUS(I,1)=ICRITP(I)
      IF(NINAG(I).EQ.0) GO TO 3020
      IDUCK=NINCL(I)
      DO 3021 J=2,IDUCK
      JJ=J-1
3021  INCLUS(I,J)=ASSGRP(I,JJ)
3020  CONTINUE
C
C        BELOW POOLS CLUSTERS FORMED FROM ASSOCIATES
C
      IA=0
3031  IA=IA+1
      IF(IA.GE.KCPB) GO TO 3030
      IF(NCLUS.EQ.1) GO TO 3030
      IDIA=NINCL(IA)
      IF(IDIA.EQ.0) GO TO 3031
      IAA=IA+1
      DO 3023 II=IAA,KCPB
      IDII=NINCL(II)
      IF(IDII.EQ.0) GO TO 3023
      DO 3025 I=1,IDIA
      DO 3025 J=1,IDII
      IF(INCLUS(II,J).EQ.INCLUS(IA,I)) GO TO 3027
3025  CONTINUE
      GO TO 3023
3027  NCLUS=NCLUS-1
      DO 3028 J=1,IDII
      DO 3029 I=1,IDIA
      IF(INCLUS(II,J).EQ.INCLUS(IA,I)) GO TO 3028
3029  CONTINUE
      NINCL(IA)=NINCL(IA)+1
      INCLUS(IA,NINCL(IA) )=INCLUS(II,J)
3028  CONTINUE
      NINCL(II)=0
      NCLINC(IA)=NCLINC(IA)+1
      CLINCL(IA,NCLINC(IA)) = II
      NCLINC(II)=0
3023  CONTINUE
      GO TO 3031
3030  CONTINUE
C
```

```
C         BELOW DESCRIBES CLUSTERS AFTER POOLING BASED ON THE ASSOCIATES
C
      WRITE(6,3033) NCLUS
 3033 FORMAT(1H0,/6X,'THERE ARE ',I3,' NONEMPTY CLUSTERS AFTER POOLING O
     *N THE BASIS OF ASSOCIATES ONLY.')
      II=0
      DO 3034 I=1,KCPB
      IF(NINCL(I).EQ.0) GO TO 3034
      II=II+1
      IDUCJ=NINCL(I)
      WRITE(6,3035)  I,(INCLUS(I,J),J=1,IDUCJ)
 3035 FORMAT(1H0,10X,'THE ACTIVITIES IN THE ',I3,'-TH CLUSTER ARE AS FOL
     *LOWS:',/,15X,50(I3,','))
 3034 CONTINUE
      DO 3101 I=1,M
      LEFT(I)=0
      DO 3102 J=1,KCPB
      IF(NINCL(J).EQ.0) GO TO 3102
      IDUCK=NINCL(J)
      DO 3110 K=1,IDUCK
      IF(I.EQ.INCLUS(J,K)) GO TO 3107
 3110 CONTINUE
 3102 CONTINUE
      GO TO 3101
 3107 LEFT(I)=J
 3101 CONTINUE
C
C         LEFTOVERS ARE ACTIVITIES NOT IN CLUSTERS AFTER ASSOCIATES HAVE
C         BEEN CONSIDERED BUT BEFORE ELIMINANTS HAVE BEEN CONSIDERED
C
C
C         DETERMINE THE NUMBER OF LEFTOVERS. NLEFT
C               LEFTO(L) = J  IMPLIES THAT THE L-TH LEFTOVER IS THE J-TH
C                             ACTIVITY
C
      NLEFT=0
      DO 3122 J=1,M
      IF(LEFT(J).NE.0) GO TO 3122
      NLEFT=NLEFT+1
      LEFTO(NLEFT)=J
 3122 CONTINUE
      WRITE(6,3123) NLEFT
 3123 FORMAT (1H0,10X,'THERE ARE ',I3,' ACTIVITIES NOT IN ANY CLUSTER YE
     *T.')
      WRITE(6,3323)THETA
 3323 FORMAT(1H1,5X,'THE ELIMINANTS OF EACH NON-CRITICAL-PATH ACTIVITY A
     *RE NOW DETERMINED FOR THETA = ',E15.5)
C
C         ELIMINANTS FOR EACH NON-CRITICAL-PATH ACTIVITY ARE NOW
C         DETERMINED
C               NNNCP = THE NUMBER OF ACTIVITIES NOT ON THE CRITICAL PATH
C               NONCP(LE) = THE LE-TH ACTIVITY NOT ON THE CRITICAL PATH
C
      NNNCP=M-KCPB
      LE=0
      DO 5000 I=1,M
      J=1
 5001 IF(I.EQ.ICRITP(J)) GO TO 5000
      J=J+1
      IF(J.LE.KCPB) GO TO 5001
 5002 LE=LE+1
```

```
      NONCP(LE)=I
5000  CONTINUE
      WRITE(6,5005) NNNCP
5005  FORMAT(1H0,10X,'THERE ARE ',I3,' ACTIVITIES NOT ON THE CRITICAL PA
     *TH.  THEY ARE AS FOLLOWS:')
      IF(NNNCP.EQ.0) GO TO 3124
      DO 5006 I=1,LE
5006  WRITE(6,5007) I,NONCP(I)
5007  FORMAT(1H ,15X,I3,'.      ',I3)
      LE=0
3126  LE=LE+1
      CTIME(IWWWQ) = COT(IWWWQ)
      CTIME(NONCP(LE)) = COT(NONCP(LE)) + THETA*SIGMA(NONCP(LE))
 7756 IWWWQ = NONCP(LE)
      DUMMY=CPTIME(CPATHT)
      CALL FPATH
3121  CONTINUE
C
C        DETERMINE THE ELIMINANTS OF THE LE-TH ACTIVITY NOT ON THE
C        CRITICAL PATH
C              NE          = THE NUMBER OF ELIMINANTS FOR THE LE-TH
C                            ACTIVITY NOT ON THE CRITICAL PATH
C              EGRP(J)       = THE J-TH ELIMINANT FOR THE LE-TH ACTIVITY
C                              NOT ON THE CRITICAL PATH
C
      NE=0
      DO 3130 K=1,KCPB
      DO 3131 I=1,KKB
      IF(IBB(I).EQ.ICRITP(K)) GO TO 3130
3131  CONTINUE
      NE=NE+1
      EGRP(NE)=ICRITP(K)
3130  CONTINUE
      WRITE(6,3133) NE,NONCP(LE)
3133  FORMAT(1H0,10X,'THERE ARE ',I3,' ELIMINANTS CORRESPONDING TO ACTIV
     *ITY ',I3)
      IF(NE.EQ.0) GO TO 3171
      DO 3135 K=1,NE
3135  WRITE(6,3136) K,NONCP(LE),EGRP(K)
3136  FORMAT(1H ,14X,'THE ',I3,'-TH ELIMINANT CORRESPONDING TO ACTIVITY
     *',I3,' IS ACTIVITY ',I3)
C
C        DETERMINE WHETHER NONCP(LE) IS AN ASSOCIATE
C              JA = 1  IF NONCP(LE)  IS AN ASSOCIATE
C              JA = 2  IF NONCP(LE)  IS NOT AN ASSOCIATE
C
      K=NONCP(LE)
      JA=1
      IF(LEFT(K).EQ.0) JA=2
      IF(JA.EQ.2) GO TO 5010
      IT=LEFT(K)
C
C        THE IT-TH CLUSTER IS EXPANDED TO INCLUDE ELIMINANTS
C
      GO TO 5011
5010  CONTINUE
C
C        ITTTT IS THE ACTIVITY NUMBER OF THE FIRST ELIMINANT
C        IT IS THE CLUSTER TO WHICH THE FIRST ELIMINANT CURRENTLY BELONG
C
      ITTT=EGRP(1)
```

```
      IT=LEFT(ITTT)
      LEFT(NONCP(LE))=IT
C
C        THE IT-TH CLUSTER IS EXPANDED TO INCLUDE ELIMINANTS
C
      NINCL(IT)=NINCL(IT)+1
      INCLUS(IT,NINCL(IT))=NONCP(LE)
      IF(NE.EQ.1) GO TO 3171
5011  DO 3172 J=JA,NE
C
C        IU IS THE ACTIVITY NUMBER OF THE NEXT ELIMINANT
C        IF IU IS IN CLUSTER K, THEN CLUSTER K IS POOLED INTO CLUSTER IT
C
      IU=EGRP(J)
      K=LEFT(IU)
3182  IF(IT.EQ.K) GO TO 3172
      NCLUS=NCLUS-1
      IW=NCLINC(K)
      DO 3183 IA=1,IW
      LEFT(ICRITP(CLINCL(K,IA)))=IT
      NCLINC(IT)=NCLINC(IT)+1
3183  CLINCL(IT,NCLINC(IT))=CLINCL(K,IA)
      NCLINC(K)=0
      IW=NINCL(K)
      NINCL(K)=0
      DO 3184 IA=1,IW
      LEFT(INCLUS(K,IA))=IT
      NINCL(IT)=NINCL(IT)+1
3184  INCLUS(IT,NINCL(IT))=INCLUS(K,IA)
3172  CONTINUE
3171  CONTINUE
      IF(LE.LT.NNNCP) GO TO 3126
C
C        END OF POOLING BASED ON ELIMINANTS EXCEPT FOR THE FOLLOWING
C        DESCRIPTION
C
      WRITE(6,3173) NCLUS
3173  FORMAT(1H1,05X,'THERE ARE ',I3,' NONEMPTY CLUSTERS AFTER POOLING O
     *N THE BASIS OF BOTH ASSOCIATES AND ELIMINANTS.')
      DO 3176 I=1,KCPB
      IF(NINCL(I).EQ.0) GO TO 3176
      IDD=NINCL(I)
      WRITE(6,3174) NINCL(I),I,(INCLUS(I,J),J=1,IDD)
3174  FORMAT(1H0,10X,'THERE ARE ',I3,' ACTIVITIES IN THE ',I3,'-TH CLUST
     *ER.  THEY ARE AS FOLLOWS:',/,20X,50(I3,','))
      IDUCK=NCLINC(I)
      WRITE(6,3175) NCLINC(I),(CLINCL(I,J),J=1,IDUCK)
3175  FORMAT(1H0,15X,I3,' CLUSTERS HAVE BEEN POOLED TO MAKE THIS CLUSTER
     *. THEY WERE AS FOLLOWS:',/,20X,50(I3,','))
3176  CONTINUE
3124  CONTINUE
      RETURN
      END
      SUBROUTINE UNION
C
C        UNION      DETERMINES UPPER AND LOWER BOUNDS ON
C        THE DISTRIBUTION FUNCTION OF THE SUBNETWORK DURATION
C        BASED ON THE UNION OF THE CLUSTERS
C
      IMPLICIT REAL*8(A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

```
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKC/ICRITP,KCPB
      DIMENSION ICRITP(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),PQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKL/FLHAT,FUHAT
      DIMENSION FLHAT(20),FUHAT(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IF(NCLUS.GT.1) GO TO 6000
      I=0
6001  I=I+1
      IF(NINCL(I).EQ.0) GO TO 6001
      NCL=I
      GO TO 6002
C*********************************************************************
C
C        POOL ALL OF THE PREVIOUS CLUSTERS INTO ONE CLUSTER
C        NCL =  THE INDEX OF THE RESULTANT POOLED CLUSTER
C        NNCL = THE NUMBER OF ACTIVITIES IN THIS POOLED CLUSTER
C
C*********************************************************************
6000  NMAX = 0
      DO 6003 I=1,KCPB
      IF(NINCL(I).LE.NMAX) GO TO 6003
      NCL =I
      NMAX=NINCL(I)
6003  CONTINUE
      DO 6004 I=1,KCPB
      IF(NINCL(I).EQ.0) GO TO 6004
      IF(I.EQ.NCL) GO TO 6004
      K=NINCL(NCL)
      JJ=NINCL(I)
      NINCL(I)=0
      DO 6005 J=1,JJ
      K=K+1
6005  INCLUS(NCL,K)=INCLUS(I,J)
      NINCL(NCL)=NINCL(NCL)+JJ
6004  CONTINUE
6002  NNCL=NINCL(NCL)
C
C        ICLGIB = -1  IF ALL CONFIGURATIONS ARE TO BE SAMPLED
C        ICLGIB = 0  IF SAMSIZ CONFIGURATIONS ARE TO BE SAMPLED USING
C                    STRATIFIED RANDOM SAMPLING
C        ICLGIB = 1  IF SAMSIZ CONFIGURATIONS ARE TO BE SAMPLED USING
```

```
C                           SIMPLE RANDOM SAMPLING
C
      IF(NNCL.LT.31) GO TO 1
      ICLGIB=1
      GO TO 2
1     IB=2**NNCL
      ICLGIB=-1
      IF(SAMSIZ.LT.IB) ICLGIB=0
2     CONTINUE
      IR=NCL
C**********************************************************************
C
C          SET  ALL ACTIVITITY DURATIONS OUTSIDE THE IR-TH CLUSTER:
C                       DISTRIBUTION
C          CTIME( ) = UPPER POINT FOR LOWER BOUND ON SUBNETWORK DURATION
C          DTIME( ) = LOWER POINT FOR UPPER BOUND ON SUBNETWORK DURATION
C                       DISTRIBUTION
C
C**********************************************************************
      DO 6006 I=1,M
      DTIME(I)=FLO(I)
6006  CTIME(I)=FHI(I)
      DO 7101 I=1,IEDF
      ULEFD(I)=0.D0
7101  LLEFD(I)=0.D0
C**********************************************************************
C
      CALL BOUND
C
C          SUBROUTINE BOUND  DETERMINES
C          EMPIRICAL DISTRIBUTION FUNCTION FOR THE SUBNETWORK DURATIONS
C          CORRESPONDING TO THE ACTIVITY DURATION CONFIGURATIONS SPECIFIED
C
C**********************************************************************
      FLHAT(1)=LLEFD(1)/SUM
      FUHAT(1)=ULEFD(1)/SUM
      DO 6900 I=2,IEDF
      II=I-1
      LLEFD(I)=LLEFD(II)+LLEFD(I)
      FLHAT(I)=LLEFD(I)/SUM
      ULEFD(I)=ULEFD(II)+ULEFD(I)
      FUHAT(I)=ULEFD(I)/SUM
6900  CONTINUE
      RETURN
      END
      SUBROUTINE SINGLE
C
C          SINGLE     DETERMINES UPPER AND LOWER BOUNDS ON
C          THE DISTRIBUTION FUNCTION OF THE SUBNETWORK DURATION
C          BASED ON THE INDIVIDUAL CLUSTERS - SEE TECHNICAL REPORT 51
C
      IMPLICIT REAL*8(A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKC/ICRITP,KCPB
      DIMENSION ICRITP(50)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
```

```
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),PQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKL/FLHAT,FUHAT
      DIMENSION FLHAT(20),FUHAT(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DO 2020 I=1,IEDF
      FLHAT(I)=0.D0
2020  FUHAT(I)=0.D0
      IR=0
3200  IR=IR+1
      IF(IR.GT.KCPB) GO TO 3208
      IF(NINCL(IR).EQ.0) GO TO 3200
      NCL=IR
      NNCL=NINCL(IR)
C
C        ICLGIB = -1  IF ALL CONFIGURATIONS ARE TO BE SAMPLED
C        ICLGIB = 0  IF SAMSIZ CONFIGURATIONS ARE TO BE SAMPLED USING
C                    STRATIFIED RANDOM SAMPLING
C        ICLGIB = 1  IF SAMSIZ CONFIGURATIONS ARE TO BE SAMPLED USING
C                    SIMPLE RANDOM SAMPLING
C
      IF(NNCL.LT.31) GO TO 1
      ICLGIB=1
      GO TO 2
1     IB=2**NNCL
      ICLGIB=-1
      IF(SAMSIZ.LT.IB) ICLGIB=0
2     CONTINUE
C*********************************************************************
C
C        SET  ALL ACTIVITITY DURATIONS OUTSIDE THE IR-TH CLUSTER:
C        CTIME( ) = UPPER POINT FOR LOWER BOUND ON SUBNETWORK DURATION
C                   DISTRIBUTION
C        CTIME( ) = LOWER POINT FOR UPPER BOUND ON SUBNETWORK DURATION
C                   DISTRIBUTION
C
C*********************************************************************
      DO 6006 I=1,M
      CTIME(I)=FLO(I)
6006  CTIME(I)=FHI(I)
      DO 7101 I=1,IEDF
      ULEFD(I)=0.D0
7101  LLEFD(I)=0.D0
C*********************************************************************
C
      CALL BOUND
C
C        SUBROUTINE BOUND  DETERMINES
C        EMPIRICAL DISTRIBUTION FUNCTION FOR THE SUBNETWORK DURATIONS
```

```
C         CORRESPONDING TO THE ACTIVITY DURATION CONFIGURATIONS SPECIFIED
C
C*****************************************************************************
      XXXX=LLEFD(1)/SUM
      IF(XXXX.GT.FLHAT(1)) FLHAT(1)=XXXX
      XXXX=ULEFD(1)/SUM
      IF(XXXX.LT.FUHAT(1)) FUHAT(1)=XXXX
      DO 6900 I=2,IEDF
      II=I-1
      LLEFD(I)=LLEFD(II)+LLEFD(I)
      XXXX=LLEFD(I)/SUM
      IF(XXXX.GT.FLHAT(I)) FLHAT(I)=XXXX
      ULEFD(I)=ULEFD(II)+ULEFD(I)
      XXXX=ULEFD(I)/SUM
      IF(XXXX.LT.FUHAT(I)) FUHAT(I)=XXXX
6900  CONTINUE
      GO TO 3200
3200  CONTINUE
      RETURN
      END
      SUBROUTINE BOUND
C
C         SUBROUTINE BOUND  DETERMINES THE
C         EMPIRICAL DISTRIBUTION FUNCTION FOR THE SUBNETWORK DURATIONS
C         CORRESPONDING TO THE ACTIVITY DURATION CONFIGURATIONS SPECIFIED
C
C         ALL ACTIVITIES OUTSIDE THE CLUSTER BEING CONSIDERED HAVE
C         FIXED DURATIONS
C         ACTIVITIES INSIDE THE CLUSTER BEING CONSIDERED HAVE BOTH
C         UPPER AND LOWER DURATIONS CONSIDERED.
C         EITHER ALL COMBINATIONS OF UPPER AND LOWER POINTS ARE
C         CONSIDERED OR A RANDOM SAMPLE THEREOF.
C
      IMPLICIT REAL*8(A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKG/FD,ULEFD,LLEFD
      REAL*8 FD(20),ULEFD(20),LLEFD(20)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKS/SUM,ICLGIB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IY=13579753
      NIB=0
      SUM=0.D0
      IF(ICLGIB) 3,6210,6211
3     CONTINUE
C*****************************************************************************
C
C         ALL ACTIVITY DURATION CONFIGURATIONS ARE TO BE CONSIDERED
C
C*****************************************************************************
2001  NIB=NIB+1
      IF(NIB.GT.IB) GO TO 2000
      CALL ALL(NIB,XL,XU,SPROB)
      X=XL-1.D-10
      I=0
6420  I=I+1
```

```
      IF(X.GT.FD(I)) GO TO 6420
      LLEFD(I) = LLEFD(I) + SPROB
      SUM=SUM+SPROB
      X=XU-1.D-10
      I=0
9420  I=I+1
      IF(X.GT.FD(I)) GO TO 9420
      ULEFD(I) = ULEFD(I) + SPROB
      GO TO 2001
6210  CONTINUE
C*********************************************************************
C
C         A STRATIFIED RANDOM SAMPLE OF THE ACTIVITY DURATION
C         CONFIGURATIONS IS TO BE CONSIDERED
C
C*********************************************************************
2011  NIB=NIB+1
      IF(NIB.GT.SAMSIZ) GO TO 2000
      CALL STRAT(NIB,XL,XU,SPROB,IY)
      X=XL-1.D-10
      I=0
7420  I=I+1
      IF(X.GT.FD(I)) GO TO 7420
      LLEFD(I) = LLEFD(I) + SPROB
      SUM=SUM+SPROB
      X=XU-1.D-10
      I=0
74201 I=I+1
      IF(X.GT.FD(I)) GO TO 74201
      ULEFD(I) = ULEFD(I) + SPROB
      GO TO 2011
6211  CONTINUE
C*********************************************************************
C
C         A SIMPLE RANDOM SAMPLE OF THE ACTIVITY DURATION
C         CONFIGURATIONS IS TO BE CONSIDERED

C*********************************************************************
3011  NIB=NIB+1
      IF(NIB.GT.SAMSIZ) GO TO 2000
      CALL SIMPLE(NIB,XL,XU,SPROB,IY)
      X=XL-1.D-10
      I=0
8420  I=I+1
      IF(X.GT.FD(I)) GO TO 8420
      LLEFD(I) = LLEFD(I) + SPROB
      SUM=SUM+SPROB
      X=XU-1.D-10
      I=0
84201 I=I+1
      IF(X.GT.FD(I)) GO TO 84201
      ULEFD(I) = ULEFD(I) + SPROB
      GO TO 3011
2000  CONTINUE
      NIB=NIB-1
      RETURN
      END
      SUBROUTINE ALL(NIB,CPTIM,CLTIM,SPROB)
C
C     ALL  RETURNS THE COMPLETION TIME AND PROBABILITY FOR THE
C     NIB-TH CONFIGURATION OF THE 2**NINCL(L) POSSIBLE CONFIGURATIONS
```

```
C
      IMPLICIT REAL*8 (A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      IJ=0
      KJ=NINCL(IR)
      CALL CONVRT(NIB,IJ,KJ,SPROB)
      CPTIM=CPTIME(CPATHT)
      CLTIM=CLTIME(CPATHT)
      RETURN
      END
      SUBROUTINE CONVRT(JP,IJ,KJ,SPROB)
      IMPLICIT REAL*8 (A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKA/CTIME,DTIME,M
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),PQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C
C     CONVERT THE INTEGER  JP  TO A BINARY NUMBER TO DEFINE THIS
C        SEGMENT OF THE ACTIVITY CONFIGURATION
C
      SPROB=1.D0
      K=JP
      DO 1 I=1,KJ
      IH=K/2
      IZ=K-IH*2
      L=INCLUS(IR,I+IJ)
      CTIME(L)=IZ*FHI(L)+(1-IZ)*FLO(L)
      DTIME(L)=CTIME(L)
      SPROB=SPROB*(IZ*PQ(L)+(1-IZ)*PP(L))
    1 K=IH
      RETURN
      END
      SUBROUTINE STRAT(NIB,CPTIM,CLTIM,SPROB,IY)
C
C        STRAT SAMPLES SYSTEMATICALLY FROM THE 2**NINCL(IR)
C        POSSIBLE CONFIGURATIONS AND COMPUTES THE COMPLETION TIME
C        AND PROBABILITY FOR EACH SAMPLED CONFIGURATION
C
C     THE ROUTINE ASSUMES THAT A SAMPLE OF SIZE  SAMSIZ  WILL BE
C        GENERATED AND RETURNS ONLY ONE SAMPLE POINT PER CALL
C
      IMPLICIT REAL*8(A-H,O-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKA/CTIME,DTIME,MM
      DIMENSION CTIME(0500),DTIME(0500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
      DIMENSION NINCL(50),INCLUS(50,100)
```

```
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKH/FHI,FLO,PP,PQ
      DIMENSION FHI(500),FLO(500),PP(500),FQ(500)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKI/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      INTEGER*4 BSUM,BCOEF
      INTEGER*2 DIGIT(31)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      SPROB=1.D0
      IF(NIB.NE.1)GO TO 10
C
C     INITIALIZE AND FIND THE FIRST CONFIGURATION
C
      M=SAMSIZ
      N=NINCL(IR)
      NM=1
      BCOEF=1
      BSUM=BCOEF
      NP1=N+1
      N1=N
      M1=M-1
      S=DFLOAT(2**N-2)/DFLOAT(M-2)
      DO 5 II=1,N
      L=INCLUS(IR,II)
      SPROB=SPROB*PQ(L)
      DTIME(L)=FHI(L)
5     CTIME(L)=FHI(L)
      CPTIM=CPTIME(CPATHT)
      CLTIM=CLTIME(CPATHT)
      CALL RAND(IY,X)
      RS=-S*X+2.D0
      GO TO 40
10    IF(NIB.GE.M)GO TO 30
C
C     FIND THE NEXT CONFIGURATION IN THE SYSTEMATIC SAMPLE
C
      RS=RS+S
      IS=IDINT(RS)
1     IF(BSUM.GE.IS)GO TO 2
      BCOEF=BCOEF*N1/(NP1-N1)
      N1=N1-1
      BSUM=BSUM+BCOEF
      GO TO 1
2     I=IS-BSUM+BCOEF
      CALL CONFIG(I,BCOEF,N1,N,DIGIT)
C
C        COMPUTE CPTIM,CLTIM,AND SPROB
C
      DO 20 II=1,N
      L=INCLUS(IR,II)
      IF(DIGIT(II))18,18,19
18    CTIME(L)=FLO(L)
      DTIME(L)=FLO(L)
      SPROB=SPROB*PP(L)
      GO TO 20
19    CTIME(L)=FHI(L)
      DTIME(L)=FHI(L)
```

```
      SPROB=SPROB*PQ(L)
20    CONTINUE
      CPTIM=CPTIME(CPATHT)
      CLTIM=CLTIME(CPATHT)
      GO TO 40
C
C     FIND THE LAST CONFIGURATION
C
30    DO 6 II=1,N
      L=INCLUS(IR,II)
      SPROB=SPROB*PP(L)
      DTIME(L)=FLO(L)
6     CTIME(L)=FLO(L)
      CPTIM=CPTIME(CPATHT)
      CLTIM=CLTIME(CPATHT)
40    RETURN
      END
      SUBROUTINE CONFIG(I,BCOEF,N1,N,DIGIT)
C
C        CONFIG  FINDS THE I-TH LARGEST N-DIGIT BINARY NUMBER
C        HAVING N1-'ONES' AND (N - N1)-'ZEROES'.  BCOEF IS THE TOTAL
C        NUMBER OF SUCH CONFIGURATIONS
C
      IMPLICIT INTEGER*4(A-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      INTEGER*2 DIGIT(31)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DO 1 II=1,N
1     DIGIT(II)=0
      N1P=N1
      NP=N
      R=I
      INDEX=N
      C=BCOEF
3     IF(N1P.LT.1)GO TO 6
      C=C*N1P/NP
      N1P=N1P-1
      NP=NP-1
      RR=R-C
      IF(RR)4,4,5
4     DIGIT(INDEX)=1
      INDEX=INDEX-1
      GO TO 3
5     INDEX=INDEX-1
      R=RR
      C=C*(NP-N1P)/NP
      NP=NP-1
      RR=R-C
      IF(RR)4,4,5
6     RETURN
      END
      SUBROUTINE SIMPLE(NIB,CPTIM,CLTIM,SPROB,IY)
      IMPLICIT REAL*8(A-H,O-Z)
C
C        SIMPLE  GENERATES (AT RANDOM) ONE OF THE 2**NINCL(IR)
C        POSSIBLE CONFIGURATIONS AND FINDS ITS COMPLETION TIME AND PROB
C
C     INITIALIZE
C
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKE/INCLUS,NINCL,NCLUS
```

```
      DIMENSION NINCL(50),INCLUS(50,100)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLK1/IEDF,NSUB,SAMSIZ
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKK/IR,IB
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      SPROB=1.D0
      IF(NIB.NE.1)GO TO 10
      NCLUS=NINCL(IR)
      NNC=NCLUS/31
      NNCR=NCLUS-NNC*31
10    IF(NNC.EQ.0)GO TO 20
      RNCL=DFLOAT(2**31-1)
      KJ=31
      DO 1 I=1,NNC
      CALL RAND(IY,U)
      JP=IDINT(U*RNCL)+1
      IF(NIB.EQ.1)JP=IDINT(RNCL+.5D0)
      IF(NIB.EQ.SAMSIZ)JP=IDINT(RNCL+1.5D0)
      IJ=(I-1)*31
      CALL CONVRT(JP,IJ,KJ,SPROB)
1     CONTINUE
20    IF(NNCR.EQ.0)GO TO 25
      RNCL=DFLOAT(2**NNCR-1)
      CALL RAND(IY,U)
      JP=IDINT(U*RNCL)+1
      IF(NIB.EQ.1)JP=IDINT(RNCL+.5D0)
      IF(NIB.EQ.SAMSIZ)JP=IDINT(RNCL+1.5D0)
      IJ=NNC*31
      CALL CONVRT(JP,IJ,NNCR,SPROB)
25    CPTIM=CPTIME(CPATHT)
      CLTIM=CLTIME(CPATHT)
      RETURN
      END
      SUBROUTINE RAND(NSEED,U)
C        ROUTINE FOR GENERATING UNIFORM RANDOM NUMBERS BETWEEN 0 AND
C      1, BASED UPON A 32 BIT LENGTH INTEGER
C
C
C      NSEED IS THE STARTING INTEGER CHOSEN FROM A RANDOM NUMBER TABLE
C      OF TEN DIGIT INTEGERS
C
C     U IS THE RETURNING UNIFORM RANDOM NUMBER
C
C
C     MD IS THE LARGEST SINGLE PRECISION INTEGER (MD = 2147483647)
C      FMD = FLOAT ( MD )
C
C      SEE HOAGLIN FOR TEST OF RANDOMNESS
C
      IMPLICIT REAL*8(A-H,O-Z)
      DATA FMD/2147483647.D0/,MD/2147483647/,ML/764261123/
      NSEED=NSEED*ML
      IF(NSEED.LT.0)NSEED=NSEED+MD+1
      U=DFLOAT(NSEED)/FMD
      RETURN
      END
      SUBROUTINE SIMPLX
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLK1/IEDF,NSUB,SAMSIZ
```

```
      INTEGER SAMSIZ
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKY/FDUHAT,FDLHAT,FHAT
      DIMENSION FDLHAT(3,20),FDUHAT(3,20),FHAT(20)
      REAL*8 FDLHAT,FDUHAT,FHAT
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLKZ/THETA
      REAL*4 THETA(3)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLK1/B1INV,OMEGA,MM,KK,MMKK,MMKK2,N,MP1,JFIRST
      COMMON /BLK2/Y1
      DIMENSION BRHS1(230),INBASE(230),XB1(230),Y1(230),B1INV(230,230)
      DIMENSION OMEGA(3)
      INTEGER Z
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      WT(OW)= (OW*2.-3.)*OW*OW+1.
C
C        THE FOLLOWING 'TOLERANCES' ARE USED IN THE ALGORITHM.
C        THEY MUST BE NON-NEGATIVE AND WOULD BE ZERO EXCEPT FOR THE
C        NUMERICAL INACCURACY OF THE COMPUTER
C              TOLR1 : IF THE MAX REDUCED COST IS LESS THAN OR EQUAL TO
C                      TOLR1 THEN ALL REDUCED COSTS ARE CONSIDERED TO BE
C                      NON-POSITIVE.
C              TOLR2 : ANY COMPONENT Y(I,J) LESS THAN OR EQUAL TO TOLR2
C                      IS CONSIDERED NON-POSITIVE.
C
      TOLR1=1.E-4
      TOLR2=1.E-4
      CALL ERRSET(208,256,-1,1)
C
C        THE INPUT
C
C        M = THE NUMBER OF CONSTRAINTS NOT INCLUDING THE OBJECTIVE
C           FUNCTION
C        N = THE NUMBER OF VARIABLES
C
      JFIRST=0
      KK=3
      MM=IEDF-1
      M=4*MM*KK+1
      N=(6*KK+7)*MM+1
      MMKK=MM*KK
      MMKK2=MMKK*2
      MP1=M+1
      DO 100 L=1,KK
100   OMEGA(L)=1./(1.+THETA(L))
C
C        BRHS IS A COLUMN OF CONSTANTS.  THE I-TH ELEMENT
C        OF BRHS IS THE CONSTANT ON THE RIGHT-HAND SIDE OF THE I-TH
C        EQUALITY IN   AX = BRHS
C
      BRHS1(1)=0.
      I1=1
      DO 610 L=1,MM
      DO 610 I=1,KK
      I1=I1+1
      I2=MMKK+I1
      I3=MMKK+I2
      I4=MMKK+I3
      BRHS1(I1)=FDUHAT(I,L)
      BRHS1(I2)=FDLHAT(I,L)
```

```
      BRHS1(I3)=BRHS1(I1)
610   BRHS1(I4)=BRHS1(I2)
      BRHS1(MP1)=1.
C
C     THE MP1-BY-N MATRIX A1 IS THE AUGMENTED MATRIX OF CONSTANTS IN
C     THE CONSTRAINTS AX = BRHS.  IT WILL BE GENERATED COLUMN-BY-
C     COLUMN AS NEEDED.
C
C
C     INBASE IS A SET OF M INTEGER VARIABLES WHICH INDICATE THE
C     COMPOSITION OF THE CURRENT BASIS B.  FOR EXAMPLE.
C          INBASE(K) = 7  IMPLIES THAT THE K-TH COLUMN IN THE BASIS B
C                    CORRESPONDS TO THE 7-TH VARIABLE
C     INBASE(1) = 0  IMPLIES THAT THE FIRST COLUMN IN THE BASIS B2
C          CORRESPONDS TO X0
C
      INBASE(1)=0
      IU=MMKK2+1
      DO 21 K=2,IU
21    INBASE(K)=K-1
      IL=IU+1
      IL=MMKK2+7*MM-1
      DO 52 K=IU,MP1
52    INBASE(K)=IL+K
C
C     THE M+1 BY M+1 MATRIX B1INV IS THE INVERSE OF THE REVISED
C     SIMPLEX BASIS MATRIX B1.
C     THE FOLLOWING STATEMENTS CONSTRUCT THE INITIAL MATRIX B1INV
C
      DO 10 II=1,MP1
      DO 12 L=1,MP1
12    B1INV(L,II)=0.
10    B1INV(II,II)=1.
      IU=MMKK2+1
      DO 11 II=2,IU
      B1INV(MMKK2+II,II)=1.
      I1=II-1
      I2=I1/KK
      L=I1-I2*KK
      IF(L.EQ.0)L=KK
11    B1INV(1,II)=-WT(OMEGA(L))
C
C     CALCULATE THE INITIAL VALUES OF THE BASIC VARIABLES
C
      DO 753 I=1,MP1
      XB1(I)=0.
      DO 755 K=1,MP1
755   XB1(I)=XB1(I)+B1INV(I,K)*BRHS1(K)
753   CONTINUE
350   CONTINUE
C
C     START THE BASIC SIMPLEX ALGORITHM
C
C     COMPUTE THE REDUCED COSTS:  C(J) -Z(J)
C     THE J-TH REDUCED COST IS DENOTED BY REDCOS(J)
C
C     FIND THE MAXIMUM REDUCED COST
C          RMAX = THE MAXIMUM REDUCED COST
C          IRMAX = THE SMALLEST INDEX J SUCH THAT REDCOS(J)=RMAX
C
      Z=1
```

```
      IRMAX=1
      RMAX=REDCST(Z,1)
      DO 24 J=2,N
      REDCOS=REDCST(Z,J)
      IF(REDCOS.LE.RMAX)GO TO 24
      RMAX=REDCOS
      IRMAX=J
24    CONTINUE
C
C        IF RMAX IS NOT POSITIVE THEN ALL THE REDUCED COSTS ARE
C        NON-POSITIVE AND THE CURRENT SOLUTION IS AN OPTIMAL SOLUTION
      IF(RMAX .LE. TOLR1) GO TO 401
C
C        DETERMINE THE IRMAX-TH Y1 VECTOR
C
      CALL YRMAX(IRMAX,MP1)
C
C        IF ALL THE COMPONENTS OF Y ARE NON-POSITIVE, THE PROBLEM HAS
C        AN UNBOUNDED SOLUTION. FOR THE PROBLEM BEING SOLVED THERE IS NO
C        NEED TO CHECK FOR UNBOUNDEDNESS SINCE CX IS BOUNDED ABOVE BY 0.
C
C        THE IRMAX-TH VARIABLE IS TO ENTER THE BASIS.
C
C        THE VARIABLE TO LEAVE THE BASIS IS NOW DETERMINED.
C             RMIN = THE MINIMUM RATIO OF XB(I)/Y(I) WITH Y(I) > 0.
C             IRMIN = THE INDEX OF THE BASIS COLUMN TO BE REMOVED
C
      RMIN=.99E20
      IRMIN=0
      DO 32 II=2,MP1
      IF(Y1(II).LE. TOLR2)  GO TO 32
      R=XB1(II)/Y1(II)
      RR=R-RMIN
      IF(RR.GE.0.)GO TO 32
      RMIN=R
      IRMIN=II
32    CONTINUE
C
C        REPLACE THE IRMIN-TH COLUMN OF THE BASIS BY THE IRMAX-TH
C        COLUMN OF A
C
C        UPDATE THE BASIS INVERSE: B1INV
C
      DO 33 J=2,MP1
      WW=B1INV(IRMIN ,J)/Y1(IRMIN )
      DO 37 L=1,MP1
37    B1INV(L,J)=B1INV(L,J)-WW*Y1(L)
33    B1INV(IRMIN ,J)=WW
C
C        UPDATE THE BASIC VARIABLES: INBASE AND XB1
C
      INBASE(IRMIN)=IRMAX
      W=XB1(IRMIN )/Y1(IRMIN )
      DO 38 I=1,MP1
38    XB1(I)=XB1(I)-Y1(I)*W
      XB1(IRMIN )=W
      GO TO 350
401    CONTINUE
999   CONTINUE
      DO 1010 L=1,MM
1010  FHAT(L)=0.D0
```

```
      DC 1011 L=1,MP1
      KKK=INBASE(L)
      IF(KKK.GT.MMKK2.AND.KKK.LE.MMKK2+MM) FHAT(KKK-MMKK2)=XB1(L  )
 1011 CONTINUE
      DO 1013 L=2,MM
 1013 FHAT(L)=FHAT(L)+FHAT(L-1)
      FHAT(IEDF)=1.D0
      RETURN
      END
      SUBROUTINE YRMAX(IRMAX,MP1)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLK2/Y1
      DIMENSION Y1(230)
      INTEGER Z
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      DO 1 Z=1,MP1
    1 Y1(Z)=-REDCST(Z,IRMAX)
      RETURN
      END
      FUNCTION REDCST(Z,I)
      IMPLICIT INTEGER*4 (C-Z)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      COMMON /BLK1/B1INV,W,M,K,MK,MK2,NN,NP1,JFIRST
      REAL*4 B1INV(230,230)
      REAL*4 W(3),WWT(3),WW(3,3),WT,OW,REDCST,DR
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
      WT(OW)= (OW*2.-3.)*OW*OW+1.
      IF(JFIRST.NE.0)GO TO 101
      JFIRST=1
      LL1=MK2
      LL2=LL1+M
      LL3=LL2+6*M
      LL4=NN
      LQ=MK2+1
      DC 100 J=1,K
      OW=W(J)
      WW(J,1)=OW
      WW(J,2)=OW*OW
      WW(J,3)=-OW*OW
  100 WWT(J)=WT(OW)
  101 IF(I.GT.LL1)GO TO 200
      P=I+1
      REDCST=-B1INV(Z,P)+B1INV(Z,MK2+P)
      IF(Z.NE.1)GO TO 500
      P=I/K
      Q=I-K*P
      IF(Q.EQ.0)Q=K
      REDCST=REDCST-WWT(Q)
      GO TO 500
  200 IF(I.GT.LL2)GO TO 300
      REDCST=0.
      Q=I-MK2
      IL=-MK+(Q-1)*K+2
      IU=1
      DC 202 P=1,4
      IL=IL+MK
      IU=IU+MK
      DC 202 L=IL,IU
  202 REDCST=REDCST-B1INV(Z,L)
      REDCST=REDCST-B1INV(Z,NP1)
      GC TO 500
```

```
300    IF(I.GT.LL3)GO TO 400
       REDCST=0.
       L=I-LL2
       P=L/3
       Q=L-3*P
       IF(Q.NE.0)GO TO 303
       Q=3
       P=P-1
303    IL=P*K+1
       IU=MK2+IL
       DO 302 R=1,K
       DR=WW(R,Q)
       IF(P.GE.M.AND.Q.EQ.1)DR=-DR
302    REDCST=REDCST-DR*(B1INV(Z,IL+R)+B1INV(Z,IU+R))
       GO TO 500
400    IF(I.GT.LL4)GO TO 500
       Q=I-LL3+1
       IF(Q-LQ)401,401,402
401    REDCST=B1INV(Z,Q)
       GO TO 500
402    REDCST=-B1INV(Z,Q)
500    RETURN
       END
```

8. SYNTH

```
C     SYNTHESIS PROGRAM
C
C  SYNTHESIS COMBINES PARALLEL AND SERIES SUBNETWORKS IDENTIFIED BY
C  THE DECOMPOSITION PROGRAM INCORPORATING SUBNETWORK DURATION
C  DISTRIBUTION INFORMATION PROVIDED BY THE SUBNETWORK ANALYSIS PROGRAM
C
C  THE FOLLOWING IS AN ALPHABETICAL LIST OF THE VARIABLES AND ARRAYS
C  USED BY BOTH THE MAIN PROGRAM AND ITS SUBROUTINES
C
C  FX(I,J)=JTH C.D.F. VALUE FOR THE ITH SUBNETWORK
C  FXSAVE(I)=ITH C.D.F. VALUE RESULTING FROM THE CURRENTLY SYNTHESIZED
C  SUBNETWORKS
C  K1=NUMBER OF THE 1ST SUBNETWORK IN PAIR CURRENTLY BEING SYNTHESIZED
C  K2=NUMBER OF THE 2ND SUBNETWORK IN PAIR CURRENTLY BEING SYNTHESIZED
C  NIEDF(I)=NUMBER OF POINTS IN THE ITH SUBNETWORK DISTRIBUTION
C  NT=NUMBER OF C.D.F. SUBDIVISIONS USED THROUGHOUT SYNTHESIS (MAX=99)
C  NUMSUB = NUMBER OF SUBNETWORKS
C  X(I,J)=JTH X VALUE FOR THE ITH SUBNETWORK
C  XSAVE(I)=ITH X VALUE RESULTING FROM THE CURRENTLY SYNTHESIZED SUBNETW
C
C  THE FOLLOWING IS AN ALPHABETICAL LIST OF THE VARIABLES AND ARRAYS USE
C  ONLY IN THE MAIN PROGRAM
C
C  ID=NUMBER OF THE SUBNETWORK
C  IEDF=NUMBER OF POINTS IN THE C.D.F.
C  INSNO(I)=ITH INSTRUCTION NUMBER
C  IOPT=0 INDICATES S WILL BE CHOSEN AS A PERCENTILE OF THE SYNTHESIZED
C         DISTRIBUTION
C      =1 INDICATES S WILL BE THE MEAN OF THE SYNTHESIZED DISTRIBUTION
C  ISORP(I)=0 INDICATES THE CURRENT SUBNETWORKS ARE IN SERIES IN THE
C             ITH INSTRUCTION
C           =1 INDICATES THE CURRENT SUBNETWORKS ARE IN PARALLEL IN THE
C             ITH INSTRUCTION
C  ISUBNT(I)=SUBNETWORK RESULTING FROM THE ITH INSTRUCTION
C  JSUBNT(I,J)=JTH SUBNETWORK CONTAINED IN THE ITH INSTRUCTION
C             =0 INDICATES THE PREVIOUS NUMBER WAS THE LAST SUBNETWORK F
C                THIS INSTRUCTION
C  NOINS=NUMBER OF INSTRUCTIONS TO BE SYNTHESIZED
C  PCT=PERCENTILE VALUE TO BE USED IN DETERMINING S
C  PD=DESIRED PROJECT DEADLINE
C  S=MEAN OR PERCENTILE OF FINAL DISTRIBUTION
C  TT=TARGET TIME USED FOR THE TIME COMPRESSION PROGRAM
C
C        FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C        NTMAX = THE MAXIMUM NUMBER OF SUBDIVIDIONS ALLOWED IN THE CDF
C        NTMAX = THE MAXIMUM NUMBER OF SUBDIVISIONS ALLOWED IN THE CDF
C                   FOR THE ENTIRE PROJECT
C        NSUBMAX = THE MAXIMUM NUMBER OF SUBNETOWRKS ALLOWED IN THE
C                   DECOMPOSITION PROCESS
C        IEDFMAX = THE MAXIMUM NUMBER OF SUBDIVISIONS ALLOWED IN THE
C                   APPROXIMATE CDF FOR EACH SUBNETWORK
C        NOINSMAX = THE MAXIMUM NUMBER OF INSTRUCTIONS GENERATED BY THE
C                   DECOMPOSITION PROCESS
C        MAX = MAXIMUM OF (NTMAX,IEDFMAX)
C        CURRENTLY, NTMAX=50; NSUBMAX=100; IEDFMAX=20; NOINSMAX=100
C
C     DIMENSION X(NSUBMAX,MAX),FX(NSUBMAX,MAX),FXSAVE(NTMAX),
C    *          NIEDF(NSUBMAX),INSNO(NOINSMAX),ISORP(NOINSMAX),
C    *          ISUBNT(NOINSMAX),JSUBNT(NOINSMAX,24),R(NTMAX),S(NTMAX),
C    *          FR(NTMAX),FS(NTMAX),FILES(NOINSMAX*27),
```

```
C       EGUIVLANECE (FILES5(NOINSMAX+1),ISUBNT(1)),
C     *             (FILES5(2*NOINSMAX+1),ISORP(1)),
C     *             FILES5(3*NOINSMAX+1),JSUBNT(1))
C
        IMPLICIT REAL*8 (A-H,O-Z)
        COMMON /BLKA/X(100,50),FX(100,50),XSAVE(50),FXSAVE(50),
      *NIEDF(100),NT
C       COMMON /BLKB/R( 50),S( 50),FR( 50),FS( 50)
        REAL*4 TTT,DPD,DCT
        DIMENSION INSNO(100),ISORP(100),ISUBNT(100),JSUBNT(100,24)
        INTEGER*4 FILES(2700)
        INTEGER                          F4/12/,F5/13/
        EGUIVALENCE (FILES(1),INSNO(1)),(FILES(101),ISUBNT(1)),
      *             (FILES(201),ISORP(1)),(FILES(301),JSUBNT(1))
C
C  READ DATA IN AND PRINT INPUT INFORMATION
C
        WRITE(6,200)
  200   FORMAT(1H1,132('*')/1X,132('*')/
      *'OTHIS IS THE OUTPUT FROM THE SYNTHESIS PROGRAM: SYNTH'/
      *1H0,132('*')/1X,132('*')//////' THE CURRENT PROJECT SCHEDULE IS TH
      *E ONE MOST RECENTLY LISTED BY THE DETERMINISTIC SCHEDULE RESOLUTIO
      *N PROGRAM.'/'OTHE PROJECT',1H','S CORRESPONDING APPROXIMATE COMPLE
      *TIOM TIME DISTRIBUTION IS DETERMINED BELOW:')
        REWIND F4
        REWIND F5
        READ(F4)NCYC,TTT,NFLAG
        READ(F5) IOPT,NT,DCT,DPD
        PCT = DCT
        TT = TTT
        PD = DPD
        NTP1=NT+1
        WRITE(6,11)NT,TT
   11   FORMAT(1H0//' FOR THIS PROBLEM'/'OTHE NUMBER OF C.D.F. SUBDIVISION
      *S USED THROUGHOUT SYNTHESIS IS',I5,'.'/' THE TARGET TIME USED BY T
      *HE DETERMINISTIC SCHEDULER WAS ',F10.5)
        READ(F5) NOINS
        IF(IOPT.EQ.0) GO TO 17
        WRITE(6,16)PD
   16   FORMAT(' THE MEAN OF THE PROJECT',1H','S APPROXIMATE COMPLETION TI
      *ME DISTRIBUTION WILL BE COMPARED TO THE'/' SPECIFIED PROJECT DEADL
      *INE TIME OF ',F12.5)
        GO TO 19
    17 P=PCT*100.
        WRITE(6,18) P,PD
   18   FORMAT(' THE ',F6.2,'-TH PERCENTILE OF THE PROJECT',1H','S APPROXI
      *MATE COMPLETION TIME DISTRIBUTION WILL BE COMPARED TO THE'/
      *' SPECIFIED PROJECT DEADLINE TIME OF ',F12.5)
    19 WRITE(6,600)
   600 FORMAT(//,      10X,'THE INSTRUCTIONS TO BE PERFORMED ARE',//)
        WRITE(6,605)
   605 FORMAT(T40,'SERIES=0')
        WRITE(6,610)
   610 FORMAT(  5X,'INSTRUCTION NO.',T25,'SUBNETWORK',T40,'PARALLEL=1',
      $T55,'SUBNETWORKS TO BE SYNTHESIZED')
        READ(F5) FILES
        DO 20 I=1,NOINS
        NSP=0
        DO 210 J=1,24
        IF(JSUBNT(I,J).LE.0)GO TO 211
  210   NSP=NSP+1
```

```
211   CONTINUE
      WRITE(6,26) INSNU(I),ISUBNT(I),ISORP(I),(JSUBNT(I,J),J=1,NSP)
   26 FORMAT(/,T11,I3,T28,I3,T43,I2,T55,24I3)
   20 CONTINUE
      WRITE(6,31)
   31 FORMAT(1H1,//,10X,'SUBNETWORK',T30,'APPROXIMATE DISTRIBUTION DETER
     $MINED USING SUBNETWORK ANALYSIS')
35    READ(F5,END=999)ID,IEDF,(X(ID,J),FX(ID,J),J=1,IEDF)
      NIEDF(ID)=IEDF
      WRITE(6,51) ID
   51 FORMAT(/,T14,I3,T34,'X',T48,'F(X)')
      WRITE(6,52) (X(ID,J),FX(ID,J),J=1,IEDF)
   52 FORMAT(  T25,F15.5,T39,F15.5)
      GO TO 35
C
C  CONSIDER INSTRUCTIONS IN REVERSE ORDER AND SYNTHESIZE ALL SUBNETWORKS
C  ACCORDING TO SERIES OR PARALLEL
C
  999 DO 100 I=1,NOINS
      II=NOINS-I+1
      DO 80 J=1,24
      K1=JSUBNT(II,J)
      JP1=J+1
      K2=JSUBNT(II,JP1)
      IF(K2.EQ.0) GO TO 75
      IF(ISORP(II).EQ.1) GO TO 55
      CALL SERIES(K1,K2)
      GO TO 60
   55 CALL PARAL(K1,K2)
C
C  STORE CURRENT DISTRIBUTION IN SUBNETWORK K2
C
   60 DO 65 K=1,NTP1
      X(K2,K)=XSAVE(K)
      FX(K2,K)=FXSAVE(K)
   65 CONTINUE
      NIEDF(K2)=NTP1
      GO TO 80
C
C  STORE DISTRIBUTION RESULTING FROM CURRENT INSTRUCTION IN SUBNETWORK
C  GIVEN BY INSTRUCTION
C
   75 DO 78 K=1,NTP1
      X(ISUBNT(II),K)=XSAVE(K)
      FX(ISUBNT(II),K)=FXSAVE(K)
   78 CONTINUE
      NIEDF(ISUBNT(II))=NTP1
      GO TO 100
   80 CONTINUE
  100 CONTINUE
C
C  OUTPUT FINAL DISTRIBUTION
C
      WRITE(6,105)
  105 FORMAT(1H1,//,10X,'THE DISTRIBUTION FOR THE SYNTHESIZED NETWORK IS
     $',//,T15,'T',T30,'F(T)')
      DO 110 K=1,NTP1
      IF(PCT.GT.FX(1,K)) KEEP=K
  110 WRITE(6,120) X(1,K),FX(1,K)
  120 FORMAT(    5X,2F15.5)
C
```

```
C  COMPUTE S AND PRODUCE INFORMATION NEEDED FOR TIME COMPRESSION
C
      IF(IOPT.EQ.0) GO TO 140
      S=X(1,1)*FX(1,1)
      DO 130 K=2,NTP1
      KM1=K-1
  130 S=S+X(1,K)*(FX(1,K)-FX(1,KM1))
      GO TO 150
  140 KEEP1=KEEP+1
      DIFF1=X(1,KEEP1)-X(1,KEEP)
      DIFF2=FX(1,KEEP1)-FX(1,KEEP)
      DIFF3=PCT-FX(1,KEEP)
      S=X(1,KEEP)+DIFF1*DIFF3/DIFF2
  150 TT=PD*TT/S
      ERRPCT=DABS(S-PD)*100./PD
      IF(IOPT)1,1,2
1     P=PCT*100.
      WRITE(6,4)P,S
4     FORMAT(//,10X,'THE ',F6.2,'-TH PERCENTILE OF THE SYNTHESIZED DISTR
     *IBUTION IS ',F12.5,/)
      GO TO 3
2     WRITE(6,5)S
5     FORMAT(//10X'THE MEAN OF THE SYNTHESIZED DISTRIBUTION IS ',F12.5/)
3     WRITE(6,180)S,ERRPCT,TT
180   FORMAT(    10X,'THE DIFFERENCE BETWEEN 'F12.5,' AND THE PROJECT DEA
     *DLINE IS ',F9.2,' PERCENT OF THE PROJECT DEADLINE.'/
     *10X,'HENCE, THE NEW TARGET TIME IS ',F12.5)
      WRITE(6,300)NCYC
300   FORMAT(//1H0,132('*')/'0THIS COMPLETES ITERATION',I7/1H0,132('*'))
      NCYC=NCYC+1
      TTT=TT
      REWIND F4
      WRITE(F4)NCYC,TTT,NFLAG
  190 STOP
      END
C
C  THE FOLLOWING IS AN ALPHABETICAL LIST OF THE VARIABLES AND ARRAYS USE
C  IN BOTH SUBROUTINES
C
C  BT=BEGINNING TIME FOR RESULTING DISTRIBUTION
C  DELT=INCREMENT FOR THE X VALUES ALLOWING FOR NT EQUAL DIVISIONS
C  FR(I)=ITH C.D.F. VALUE FOR THE 1ST SUBNETWORK OF PAIR TO BE SYNTHESIZ
C  FS(I)=ITH C.D.F. VALUE FOR THE 2ND SUBNETWORK OF PAIR TO BE SYNTHESIZ
C  NR=NUMBER OF POINTS IN 1ST SUBNETWORK
C  NS=NUMBER OF POINTS IN 2ND SUBNETWORK
C  R(I)=ITH X VALUE FOR THE 1ST SUBNETWORK OF PAIR TO BE SYNTHESIZED
C  S(I)=ITH X VALUE FOR THE 2ND SUBNETWORK OF PAIR TO BE SYNTHESIZED
C  T=X VALUE FOR THE RESULTING SYNTHESIZED SUBNETWORK
C  XMAX=MAXIMUM POSSIBLE X VALUE FOR THE RESULTING SYNTHESIZED SUBNETWOR
C  XMIN=MINIMUM POSSIBLE X VALUE FOR THE RESULTING SYNTHESIZED SUBNETWOR
C
      SUBROUTINE PARAL(K1,K2)
C
C  SYNTHESIZES 2 PARALLEL SUBNETWORKS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/X(100,50),FX(100,50),XSAVE(50),FXSAVE(50),
     *NIEDF(100),NT
      COMMON /BLKB/R( 50),S( 50),FR( 50),FS( 50)
      NR=NIEDF(K1)
      NS=NIEDF(K2)
```

```
      DO 10 I=1,NR
      R(I)=X(K1,I)
   10 FR(I)=FX(K1,I)
      DO 15 I=1,NS
      S(I)=X(K2,I)
   15 FS(I)=FX(K2,I)
      XMIN=DMIN1(R(1),S(1))
      XMAX=DMAX1(R(NR),S(NS))
      BT=XMIN
      DELT=(XMAX-XMIN)/NT
      T=BT-DELT
      DO 50 I=1,NT
      F1=0.0D0
      F2=0.0D0
      T=T+DELT
      DO 40 J=1,NR
      IF(R(J).GT.T) GO TO 41
      F1=FR(J)
   40 CONTINUE
   41 DO 45 J=1,NS
      IF(S(J).GT.T) GO TO 46
      F2=FS(J)
   45 CONTINUE
   46 FXSAVE(I)=F1*F2
      XSAVE(I)=T
   50 CONTINUE
      NTP1=NT+1
      FXSAVE(NTP1)=1.0D0
      XSAVE(NTP1)=XMAX
      RETURN
      END
      SUBROUTINE SERIES(K1,K2)
C
C  SYNTHESIZES 2 SUBNETWORKS IN SERIES
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLKA/X(100,50),FX(100,50),XSAVE(50),FXSAVE(50),
     *NIEDF(100),NT
      COMMON /BLKB/R( 50),S( 50),FR( 50),FS( 50)
      NR=NIEDF(K1)
      NS=NIEDF(K2)
      DO 10 I=1,NR
      R(I)=X(K1,I)
   10 FR(I)=FX(K1,I)
      DO 15 I=1,NS
      S(I)=X(K2,I)
   15 FS(I)=FX(K2,I)
      XMIN=R(1)+S(1)
      XMAX=R(NR)+S(NS)
      BT=XMIN
      DELT=(XMAX-XMIN)/NT
      T=BT-DELT
      DO 55 I=1,NT
      FT=0.0D0
      T=T+DELT
      DO 50 J=1,NS
      IF(S(J).GT.T) GO TO 54
      IF(J.GT.1) GO TO 30
      FF=FS(1)
      GO TO 35
   30 JJ=J-1
```

```
      FF=FS(J)-FS(JJ)
   35 FTMS=0.0D0
      TMS=T-S(J)
      DO 40 K=1,NR
      IF(R(K).GT.TMS) GO TO 45
      FTMS=FR(K)
   40 CONTINUE
   45 FT=FT+FF*FTMS
   50 CONTINUE
   54 XSAVE(I)=T
      FXSAVE(I)=FT
   55 CONTINUE
      NTP1=NT+1
      FXSAVE(NTP1)=1.0D0
      XSAVE(NTP1)=XMAX
      RETURN
      END
```

9. SAVEFIL

```
C       PROGRAM SAVEFIL
C
C          THIS PROGRAM TRANSFERS THE OPERATIONAL INFORMATION GENERATED
C          BY THE STATISTICAL PERT PROGRAMS FROM THE TEMPORARY FILES TO
C          THE PERMANENT DATA SET DEFINED AS I/O-UNIT PS
C
C
C          INPUT INSTRUCTIONS
C
C          A SINGLE CARD SHOULD BE SUPPLIED AS INPUT TO THIS PROGRAM.
C          THE CARD SHOULD BEAR IN COLUMNS 1 - 8 THE NAME OF THE LAST
C          PROGRAM EXECUTED BEFORE THIS INTERRUPT.
C          (FORMAT 2A4, LEFT JUSTIFIED)
C
C          FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C          MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                      PROJECT NETWORK
C          BKMAX = THE MAXIMUM NUMBER OF BREAK POINTS ALLOWED IN THE
C                      ENTIRE PROJECT'S TIME-COST CURVE
C          NCTMAX = THE MAXIMUM NUMBER OF COMPLETION TIMES AND COSTS FOR
C                      EACH ACTIVITY
C          IEDFMAX = THE MAXIMUM NUMBER OF SUBDIVISIONS ALLOWED IN THE
C                      APPROXIMATE CDF FOR EACH SUBNETWORK
C          NOINSMAX = THE MAXIMUM NUMBER OF INSTRUCTIONS GENERATED BY THE
C                      DECOMPOSITION PROCESS
C
C       INTEGER*4 FILE0(4*MMAX+5),NCT(MMAX),FILE3B(3*MMAX+5),
C      *        FILE5(NOINSMAX*27)
C       INTEGER*2 BREAK(BKMAX),TIMCST(2*NCTMAX),FILE6(2*MMAX)
C       REAL*8 FD(IEDFMAX),X(IEDFMAX)
C       EQUIVALENCE (FILE0(3*MMAX+6),NCT(1))
C
C
        INTEGER F0/8/,F1/9/,F2/10/,F3/11/,F4/12/,F5/13/,F6/14/,F7/15/
        INTEGER PS/4/
        INTEGER FILE0(4005),NCT(1000),FILE3A(11),FILE3B(3005),FILE5(2700),
       *       LIST(24)
        REAL FILE1(5),FILE2(5)
        REAL*8 CDF(12),FD(20),X(20)
        INTEGER*2 NBREAK,BREAK(3000),TIMCST(12),TEST1,TEST2,TEST3,STIME,
       *        FILE6(2000)
        EQUIVALENCE (FILE0(1),NACT),(FILE0(3006),NCT(1))
        EQUIVALENCE (FILE1(1),FILE2(1),FILE3A(1),FILE3B(1),FILE5(1),
       *          FILE6(1),BREAK(1),TIMCST(1),LIST(1),CDF(1),FD(1))
        INTEGER IPROG(2)
        INTEGER PROG(13)/'MAIN','LOOP','DPS ','DSR ','SIMP','MODS','DECO',
       *                'SUBN','SYNT','DSR ','MODS','SUBN','SYNT'/
C
        WRITE(6,112)
        REWIND F4
        READ(F4)NCYC
        READ(5,110)IPROG
        K1=1
        IF(NCYC.GE.2)K1=10
        DO 1 I=K1,13
        IF(IPROG(1).EQ.PROG(I))GO TO 2
1       CONTINUE
        WRITE(6,111)
        GO TO 109
2       K=I
```

```
      REWIND F0
      REWIND F1
      REWIND F2
      REWIND F3
      REWIND F4
      REWIND F5
      REWIND F6
      REWIND F7
      REWIND PS
C
C        READ AND STORE FILE0
C
      READ(F0) FILE0
      WRITE(PS)FILE0
      READ(F0) JMAT
      WRITE(PS)JMAT
      IF(K.LT.3)GO TO 3
      READ(F0) NBREAK,(BREAK(I),I=1,NBREAK)
      WRITE(PS)NBREAK,(BREAK(I),I=1,NBREAK)
3     CONTINUE
C
C        READ AND STORE FILE1
C
      READ(F1) TEST1,TEST2,TEST3,STIME
      WRITE(PS)TEST1,TEST2,TEST3,STIME
      DO 4 I=1,NACT
      NNCT=NCT(I)
      NNCT1=NNCT-1
      NNCT2=2*NNCT
      READ(F1) (TIMCST(J),J=1,NNCT2)
      WRITE(PS) (TIMCST(J),J=1,NNCT2)
      DO 4 J=1,NNCT1
      READ(F1) FILE1
      WRITE(PS)FILE1
4     CONTINUE
C
C        READ AND STORE FILE2
C
      IF(K.NE.4.AND.K.NE.5.AND.K.NE.10)GO TO 6
      DO 5 I=1,NACT
      READ(F2) FILE2
      WRITE(PS)FILE2
5     CONTINUE
6     CONTINUE
C
C        READ AND STORE FILE3
C
      READ(F3) FILE3A
      WRITE(PS)FILE3A
      IF(K.NE.5.AND.K.NE.6)GO TO 7
      READ(F3) FILE3B
      WRITE(PS)FILE3B
7     CONTINUE
      IF(K.LT.7)GO TO 10
8     READ(F3,END=9)M
      WRITE(PS)      M
      MM=2+3*M
      READ(F3) (FILE3B(I),I=1,MM)
      WRITE(PS)(FILE3B(I),I=1,MM)
      GO TO 8
9     M=0
```

AD-A060 568 TEXAS A AND M UNIV COLLEGE STATI.N INST OF STATISTICS F/G 9/2
A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJEC--ETC(U)
AUG 78 T C BAKER, R L SIELKEN N00014-78-C-0426
UNCLASSIFIED THEMIS-TR-57 NL

4 OF 4
AD
A060568

END
DATE
FILMED
01 -79
DDC

```
      WRITE(PS)M
10    CONTINUE
C
C        READ AND STORE FILE4
C
      READ(F4) NCYC,TT,NFLAG
      WRITE(PS)NCYC,TT,NFLAG
      IF(K.NE.6.AND.K.NE.7.AND.K.NE.11)GO TO 12
      READ(F4) JNACT
      WRITE(PS)JNACT
      DO 11 I=1,JNACT
      READ(F4) CDF
      WRITE(PS)CDF
11    CONTINUE
12    CONTINUE
C
C        READ AND STORE FILES5
C
      READ(F5) IOPT,NT,PCT,PD
      WRITE(PS)IOPT,NT,PCT,PD
      IF(K.LT.7)GO TO 15
      READ(F5) NOINS
      WRITE(PS)NOINS
      READ(F5) FILES
      WRITE(PS)FILES
      IF(K.NE.8.AND.K.NE.12)GO TO 15
13    READ(F5,END=14)ID,IEDF,(FD(I),X(I),I=1,IEDF)
      WRITE(PS)       ID,IEDF,(FD(I),X(I),I=1,IEDF)
      GO TO 13
14    ID=-1
      WRITE(PS)       ID,IEDF,(FD(I),X(I),I=1,IEDF)
15    CONTINUE
C
C     READ AND STORE FILE6
C
      IF(K.LT.3)GO TO 17
      NACT2=2*NACT
      DO 16 I=1,NBREAK
      READ(F6) (FILE6(J),J=1,NACT2)
      WRITE(PS)(FILE6(J),J=1,NACT2)
16    CONTINUE
17    CONTINUE
C
C        READ AND STORE FILE7
C
      IF(K.LT.5)GO TO 20
18    READ(F7,END=19)LIST
      WRITE(PS)       LIST
      GO TO 18
19    LIST(1)=-1
      WRITE(PS)LIST
20    CONTINUE
C
C     WRITE TERMINATION INFORMATION
C
      WRITE(6,113)PS,NCYC,IPROG
109   STOP
110   FORMAT(2A4)
111   FORMAT(1H0,5X,'THE NAME SPECIFIED FOR THE LAST PROGRAM EXECUTED IS
     * INVALID.')
112   FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FOR THE SYSTE
```

```
     *M INTERRUPTION PROGRAM: SAVEFIL'/1H0,132('*')/1X,132('*'))
 113  FORMAT(1H0,5X,'THE OPERATIONAL INFORMATION GENERATED BY THE STATIS
     *TICAL PERT PROGRAMS HAS BEEN SUCCESSFULLY TRANSFERRED'/6X,'TO THE
     *DATA SET DEFINED AS I/O-UNIT ',I2//6X,'THE PROCEDURE WAS INTERRUPT
     *ED DURING THE ',I5,'-TH ITERATION.'/6X,2A4,' WAS THE LAST PROGRAM
     *EXECUTED.')
      END
```

10. READFIL

```
C      PROGRAM READFIL
C
C         THIS PROGRAM RETRIEVES THE OPERATIONAL INFORMATION FOR THE
C         STATISTICAL PERT PROGRAMS FROM THE PERMANENT FILE REFERENCED BY
C         I/O-UNIT PS AND RESTORES IT TO THE APPROPRIATE TEMPORARY FILES
C
C
C         INPUT INSTRUCTIONS
C
C         A SINGLE CARD SHOULD BE SUPPLIED AS INPUT TO THIS PROGRAM.
C
C         THE CARD SHOULD BEAR THE FOLLOWING:
C               COLS  1- 8:  THE NAME OF THE LAST PROGRAM EXECUTED BEFORE
C                            THE INTERRUPT (FORMAT 2A4, LEFT JUSTIFIED)
C               COL      10:  0  IF THE PROCEDURE WAS INTERRPUTED BEFORE
C                                THE FIRST EXECUTION OF SYNTH
C                             1  OTHERWISE
C               THE REMAINING PARAMETERS SHOULD ONLY BE SUPPLIED IF THE
C               PROCEDURE WAS INTERRUPTED IMMEDIATELY AFTER THE EXECUTION
C               OF SYNTH
C
C               COLS 11-20:  THE VALUE OF  TT  FOR THE NEXT ITERATION IF
C                            DIFFERENT FROM THE VALUE SPECIFIED BY SYNTH
C               COLS 21-25:  1  IF THE VALUES OF ANY OF THE FOLLOWING
C                               PARAMETERS ARE TO BE DIFFERENT FROM THEIR
C                               CURRENT VALUES IN THE NEXT ITERATION:
C                                    IOPT,PCT,PD
C                            0  OTHERWISE
C               COLS 26-30:  THE VALUE OF  IOPT  FOR THE NEXT ITERATION
C               COLS 31-40:  THE VALUE OF  PCT  FOR THE NEXT ITERATION
C               COLS 41-50:  THE VALUE OF  PD  FOR THE NEXT ITERATION
C
C         FOR THE SAKE OF IDENTIFYING THE APPROPRIATE DIMENSIONS, LET
C         MMAX = THE MAXIMUM NUMBER OF ACTIVITIES IN THE ORIGINAL
C                   PROJECT NETWORK
C         BKMAX = THE MAXIMUM NUMBER OF BREAK POINTS ALLOWED IN THE
C                   ENTIRE PROJECT'S TIME-COST CURVE
C         NCTMAX = THE MAXIMUM NUMBER OF COMPLETION TIMES AND COSTS FOR
C                   EACH ACTIVITY
C         IEDFMAX = THE MAXIMUM NUMBER OF SUBDIVISIONS ALLOWED IN THE
C                   APPROXIMATE CDF FOR EACH SUBNETWORK
C         NOINSMAX = THE MAXIMUM NUMBER OF INSTRUCTIONS GENERATED BY THE
C                   DECOMPOSITION PROCESS
C         CURRENTLY, MMAX=1000; BKMAX=3000; NCTMAX=6; IEDFMAX=20;
C                   NOINSMAX=100
C
C      INTEGER*4 FILE0(4*MMAX+5),NCT(MMAX),FILE3B(3*MMAX+5),
C     *          FILE5(NOINSMAX*27)
C      INTEGER*2 BREAK(BKMAX),TIMCST(2*NCTMAX),FILE6(2*MMAX)
C      REAL*8 FD(IEDFMAX),X(IEDFMAX)
C      EQUIVALENCE (FILE0(3*MMAX+6),NCT(1))
C
       INTEGER F0/8/,F1/9/,F2/10/,F3/11/,F4/12/,F5/13/,F6/14/,F7/15/
       INTEGER PS/4/
       INTEGER FILE0(4005),NCT(1000),FILE3A(11),FILE3B(3005),FILE5(2700),
      *        LIST(24)
       REAL FILE1(5),FILE2(5)
       REAL*8 CDF(12),FD(20),X(20)
       INTEGER*2 NBREAK,BREAK(3000),TIMCST(12),TEST1,TEST2,TEST3,STIME,
      *          FILE6(2000)
```

```
      EQUIVALENCE (FILE0(1),NACT),(FILE0(3006),NCT(1))
      EQUIVALENCE (FILE1(1),FILE2(1),FILE3A(1),FILE3B(1),FILE5(1),
     *            FILE6(1),BREAK(1),TIMCST(1),LIST(1),CDF(1),FD(1))
      INTEGER IPROG(2)
      INTEGER PROG(13)/'MAIN','LOOP','DPS ','DSR ','SIMP','MODS','DECO',
     *                'SUBN','SYNT','DSR ','MODS','SUBN','SYNT'/
C
      WRITE(6,112)
      READ(5,110)IPROG,NCYC,IFLG,TTT,IIOPT,PPCT,PPD
      K1=10
      IF(NCYC.EQ.0)K1=1
      DO 1 I=K1,13
      IF(IPROG(1).EQ.PROG(I))GO TO 2
    1 CONTINUE
      WRITE(6,111)
      GO TO 109
    2 K=I
      REWIND F0
      REWIND F1
      REWIND F2
      REWIND F3
      REWIND F4
      REWIND F5
      REWIND F6
      REWIND F7
      REWIND PS
C
C       RESTORE FILE0
C
      READ(PS) FILE0
      WRITE(F0)FILE0
      READ(PS) JMAT
      WRITE(F0)JMAT
      IF(K.LT.3)GO TO 3
      READ(PS) NBREAK,(BREAK(I),I=1,NBREAK)
      WRITE(F0)NBREAK,(BREAK(I),I=1,NBREAK)
      ENDFILE F0
    3 CONTINUE
C
C       RESTORE FILE1
C
      READ(PS) TEST1,TEST2,TEST3,STIME
      WRITE(F1)TEST1,TEST2,TEST3,STIME
      DO 4 I=1,NACT
      NNCT=NCT(I)
      NNCT1=NNCT-1
      NNCT2=2*NNCT
      READ(PS) (TIMCST(J),J=1,NNCT2)
      WRITE(F1)(TIMCST(J),J=1,NNCT2)
      DO 4 J=1,NNCT1
      READ(PS) FILE1
      WRITE(F1)FILE1
    4 CONTINUE
      ENDFILE F1
C
C       RESTORE FILE2
C
      IF(K.NE.4.AND.K.NE.5.AND.K.NE.10)GO TO 6
      DO 5 I=1,NACT
      READ(PS) FILE2
      WRITE(F2)FILE2
```

```
5       CONTINUE
        ENDFILE F2
6       CONTINUE
C
C          RESTORE FILE3
C
        READ(PS) FILE3A
        WRITE(F3)FILE3A
        IF(K.NE.5.AND.K.NE.6)GO TO 7
        READ(PS) FILE3B
        WRITE(F3)FILE3B
7       CONTINUE
        IF(K.LT.7)GO TO 10
8       READ(PS)M
        IF(M.EQ.0)GO TO 9
        WRITE(F3)M
        MM=2+3*M
        READ(PS) (FILE3B(I),I=1,MM)
        WRITE(F3)(FILE3B(I),I=1,MM)
        GO TO 8
9       CONTINUE
        ENDFILE F3
10      CONTINUE
C
C          RESTORE FILE4
C
        READ(PS) NCYC,TT,NFLAG
        IF(TTT.NE.0.0)TT=TTT
        WRITE(F4)NCYC,TT,NFLAG
        IF(K.NE.6.AND.K.NE.7.AND.K.NE.11)GO TO 12
        READ(PS) JNACT
        WRITE(F4)JNACT
        DO 11 I=1,JNACT
        READ(PS) CDF
        WRITE(F4)CDF
11      CONTINUE
12      CONTINUE
C
C          RESTORE FILE5
C
        READ(PS) IOPT,NT,PCT,PD
        IF(IFLG.EQ.0)GO TO 21
        IOPT=IIOPT
        PCT=PPCT
        PD=PPD
21      WRITE(F5)IOPT,NT,PCT,PD
        IF(K.LT.7)GO TO 15
        READ(PS) NOINS
        WRITE(F5)NOINS
        READ(PS) FILE5
        WRITE(F5)FILE5
        IF(K.NE.8.AND.K.NE.12)GO TO 15
13      READ(PS)        ID,IEDF,(FD(I),X(I),I=1,IEDF)
        IF(ID.EQ.-1)GO TO 14
        WRITE(F5)       ID,IEDF,(FD(I),X(I),I=1,IEDF)
        GO TO 13
14      CONTINUE
        ENDFILE F5
15      CONTINUE
C
C          RESTORE FILE6
```

```
C
      IF(K.LT.3)GO TO 17
      NACT2=2*NACT
      DO 16 I=1,NBREAK
      READ(PS) (FILE6(J),J=1,NACT2)
      WRITE(F6)(FILE6(J),J=1,NACT2)
16    CONTINUE
      ENDFILE F6
17    CONTINUE
C
C        RESTORE FILE7
C
      IF(K.LT.5)GO TO 20
18    READ(PS)      LIST
      IF(LIST(1).EQ.-1)GO TO 19
      WRITE(F7)     LIST
      GO TO 18
19    CONTINUE
      ENDFILE F7
20    CONTINUE
C
C     WRITE TERMINATION INFORMATION
C
      WRITE(6,113)    NCYC,IPROG
109   STOP
110   FORMAT(2A4,I2,F10.5,2I5,2F10.5)
111   FORMAT(1H0,5X,'THE NAME SPECIFIED FOR THE LAST PROGRAM EXECUTED IS
     * INVALID.')
112   FORMAT(1H1,132('*')/1X,132('*')/'0THIS IS THE OUTPUT FOR THE SYSTE
     *M RESTART PROGRAM: READFIL'/1H0,132('*')/1X,132('*'))
113   FORMAT(1H0,5X,'THE OPERATIONAL INFORMATION GENERATED BY THE STATIS
     *TICAL PERT PROGRAMS HAS BEEN SUCCESSFULLY RESTORED'/6X,   'TO THE
     *TEMPORARY DATA SETS.',            //6X,'THE PROCEDURE WAS INTERRUPT
     *ED DURING THE ',I5,'-TH ITERATION.'/6X,2A4,' WAS THE LAST PROGRAM
     *EXECUTED.')
      END
```

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Texas A&M University	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP Unclassified

3. REPORT TITLE

A USER'S GUIDE TO THE COMPUTER IMPLEMENTATION OF THE NEW PROJECT SCHEDULING PROCEDURE: STATISTICAL PERT

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report

5. AUTHOR(S) (First name, middle initial, last name)

Thomas C. Baker, Jr. and Robert L. Sielken Jr.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
August 1978	301	12

8a. CONTRACT OR GRANT NO. N00014-78-C-0426	9a. ORIGINATOR'S REPORT NUMBER(S) Report No. 57
b. PROJECT NO. NR047-179	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

This report documents a new project scheduling procedure developed at the which Institute of Statistics, Texas A&M University. The project scheduling algorithm is a five step iterative procedure capable of determining a minimum cost project schedule when the activities making up the project have durations which are random variables. The cost of an activity is assumed to be a convex piecewise linear function of the activity's mean duration. The problem is to determine the activity mean durations which both minimize the total project cost and insure that the mean (or some specified percentile) of the corresponding project completion time distribution is less than or equal to a specified project deadline. The entire distribution of the project's completion time under the minimum cost schedule is a valuable by-product. Information on the trade-off between the project's minimum cost and its specified deadline is also provided.

This report is intended as a user's guide to the new project scheduling procedure and its computer implementation. The report includes a description of the project scheduling problem, a general overview of the scheduling procedure including references to technical reports documenting the development of the procedure, and an example of the procedure's performance. The documentation of the computer implementation includes specific input instructions; sample input and output; flowcharts; individual program descriptions; technical details concerning temporary data sets, job control language, and program interruption and restart procedures; and program listings.

ATTACHMENT III

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

Unclassified

Security Classification

Unclassified

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Statistical PERT						
Project Scheduling						
Minimum Cost Scheduling						
Random Activity Durations						
Project Completion Time Distribution						
Network Duration Distribution						
User's Guide						
Computer Programs for PERT						

ATTACHMENT III (Continued)

DD FORM 1 NOV 65 1473 (BACK)

(PAGE 2)

Unclassified

BASIC DISTRIBUTION LIST FOR
UNCLASSIFIED TECHNICAL REPORTS

	Copies
Operations Research Office of Naval Research (Code 434) Arlington, Virginia 22217	3
Director, Naval Research Laboratory Attn: Library, Code 2029 (ONRL) Washington, D.C. 20390	2
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12
Defense Logistics Studies Information Exchange Army Logistics Management Center Fort Lee, Virginia 23801	1
Technical Information Center Naval Research Laboratory Washington, D.C. 20390	6
Office of Naval Research New York Area Office 207 West 24th Street New York, New York 10011 Attn: Dr. J. Laderman	1
Director, Office of Naval Research Branch Office 1030 East Green Street Pasadena, California 91101 Attn: Dr. A.R. Laufer	1
Yale University Department of Statistics Box 2179 - Yale Station New Haven, CT 06520 Attn: Prof. I.R. Savage	1
Mr. William Dejka Code 360 Naval Electronics Laboratory Center San Diego, California 92132	1
Professor Lucien A. Schmit, Jr. Room 6731, Boelter Hall School of Engrng. & Applied Sci. University of California Los Angeles, California 90024	1

	Copies
Director Office of Naval Research Branch Office 495 Summer Street Boston, Massachusetts 02210 Attn: Dr. A.L. Powell	1
Prof. Emanuel Parzen Institute of Statistics Texas A&M University College Station, TX 77843	1
Office of Naval Research San Francisco Area Office 760 Market St. - Room 447 San Francisco, California 94103	1
Technical Library Naval Ordnance Station Indian Head, Maryland 20640	1
Bureau of Naval Personnel Navy Department Technical Library Washington, D.C. 20370 STOP 82	2
Library, Code 0212 Naval Postgraduate School Monterey, California 93940	1
Naval Ordnance Station Louisville, Kentucky 40214	1
Library Naval Electronics Laboratory Center San Diego, California 92152	1
Naval Ship Engineering Center Philadelphia Division Technical Library Philadelphia, Pennsylvania 19112	1
Dr. A.L. Slafkosky Scientific Advisor Commandant Marine Corps (Code AX) Washington, D.C. 20380	1

	Copies
Purdue University Graduate School of Ind. Admin. Lafayette, Indiana 47907 Attn: Prof. Andrew Whinston	1
Department of Economics Tisch Hall - 5th Floor Washington Square, NY Univ. New York City, NY 10003 Attn: Prof. O. Morgenstern	1
Dept. of Statistics Syntex Research 3401 Hillview Palo Alto, CA 94304 Attn: Prof. Stuart Bessler	1
University of California Department of Engineering Los Angeles, California 90024 Attn: Prof. R.R. O'Neill	1
Maritime Transportation Research Board National Academy of Sciences 2101 Constitution Avenue Washington, D.C. 20418 Attn: RADM J.B. Oren USCG (RET)	1
Stanford University Department of Operations Research Stanford, California 94305 Attn: Prof. A.F. Veinott, Jr.	1
Texas A&M Foundation College Station, Texas 77843 Attn: Prof. H.O. Hartley	1
Case Western Reserve University Cleveland, Ohio 44106 Attn: Prof. B.V. Dean	1
University of California Center for Research in Management Science Berkeley, California 94720 Attn: Prof. W.I. Zangwill	1
U.S. Naval Postgraduate School Department of Operations Research and Economics Monterey, California 93940 Attn: Prof. C.R. Jones	

	Copies
Dr. Del Cilmer Naval Weapons Center (Code 607) China Lake, California 93555	1
Dr. Paul Murrill Project Themis Department of Chemical Engineering Louisiana State University Baton Rouge, Louisiana 70803	1
Director Army Materials & Mechanics Research Center Attn: Mr. J. Bluhm Watertown, Massachusetts 02172	1
Commanding Officer U.S. Army Ballistic Research Laboratories Attn: Dr. John H. Giese Aberdeen Proving Ground, Maryland 21005	1
University of Iowa Department of Mechanics and Hydraulics Iowa City, Iowa 52240 Attn: Prof. W.F. Ames	1
Office of Naval Research Resident Representative 1105 Guadalupe Lowich Building Austin, Texas 78701 Attn: Mr. Francis M. Lucas	1
Dr. Jerome Bracken Institute of Defense Analyses 400 Army-Navy Drive Arlington, Virginia 22202	1
Professor Richard L. Fox School of Engineering Case Western Reserve University Cleveland, Ohio 44106	1
	1

Copies

1

1

University of Chicago
Statistical Research Center
Chicago, Illinois 60637
Attn: Prof. W. Kruskal — 1

Stanford University
Department of Statistics
Stanford, California 94305
Attn: Prof. G.J. Lieberman — 1

1

Princeton University
Department of Mathematics
Princeton, New Jersey 08540
Attn: Prof. J.W. Tukey — 1

Stanford University
Department of Statistics
Stanford, California 94305
Attn: Prof. T.W. Anderson — 1

University of California
Department of Statistics
Berkeley, California 94720
Attn: Prof. P.J. Bickel — 1

University of Washington
Department of Mathematics
Seattle, Washington 98105
Attn: Prof. Z.W. Birnbaum — 1

Cornell University
Department of Mathematics
White Hall
Ithaca, New York 14850
Attn: Prof. J. Kiefer — 1

Harvard University — Copies
Department of Statistics
Cambridge, Massachusetts 02139
Attn: Prof. W.G. Cochran — 1

Columbia University
Department of Industrial Engineering
New York, New York 10027
Attn: Prof. C. Derman — 1

New York University
Institute of Mathematical Sciences
New York, New York 10453
Attn: Prof. W.M. Hirsch — 1

University of North Carolina
Statistics Department
Chapel Hill, North Carolina 27515
Attn: Prof. W.L. Smith and
Prof. M.R. Leadbetter — 1

Purdue University
Division of Mathematical Sciences
Lafayette, Indiana 47079
Attn: Prof. H. Rubin — 1

University of California, San Diego
Department of Mathematics
P.O. Box 109
La Jolla, California 92038
Attn: Prof. M. Rosenblatt — 1

Florida State University
Department of Statistics
Tallahassee, Florida 32306
Attn: Prof. R.A. Bradley — 1

New York University
Department of Industrial Engineering & Operations Research
Bronx, New York 10453
Attn: Prof. J.H.K. Kao — 1

University of Wisconsin
Department of Statistics
Madison, Wisconsin 53706
Attn: Prof. G.E.P. Box — 1

Logistics Research Project The George Washington University 707 - 22nd Street, N.W. Washington, D. C. 20037 Attn: Dr. W. H. Marlow	Copies 1
International Business Machines Corporation P.O. Box 218, Lamb Estate Yorktown Heights, New York 10598 Attn: Dr. Alan Hoffman	1
University of California Management Sciences Research Project Los Angeles, California 90024 Attn: Dr. J.R. Jackson	1
Harvard University Department of Economics Cambridge, Massachusetts 02138 Attn: Prof. K.J. Arrow	1
Cowles Commission for Research in Economics Yale University New Haven, Connecticut 06520 Attn: Prof. Martin Shubik	1
Carnegie-Mellon University Graduate School of Industrial Administration Pittsburgh, Pennsylvania 15213 Attn: Prof. G. Thompson	1
University of California Department of Economics Berkeley, California 94720 Attn: Prof. R. Radner	1
University of California Operations Research Center Institute of Engineering Research Berkeley, California 94720 Attn: Prof. D. Gale	1
University of California Graduate School of Business Administration Los Angeles, California 90024 Attn: Prof. J. Marschak	1

The University of Michigan Department of Mathematics, W.E. Ann Arbor, Michigan 48104 Attn: Prof. R.M. Thrall	Copies 1
Princeton University Department of Mathematics Princeton, New Jersey 08540 Attn: Prof. A.W. Tucker	1
Case Western Reserve University Systems Research Center Cleveland, Ohio 44106 Attn: Prof. M. Mesarovic	1
University of Texas Department of Mathematics Austin, Texas 78712 Attn: Dr. A. Charnes	1
Stanford University Department of Operations Research Stanford, California 94305 Attn: Dr. D.L. Iglehart	1
University of Delaware Department of Mathematics Newark, Delaware 19711 Attn: Prof. R. Remage, Jr.	1
Stanford University Department of Operations Research Stanford, California 94305 Attn: Prof. F.S. Hillier	1
Dr. Claude-Alain Burdet Asst. Prof. Industrial Admin. Carnegie-Mellon University Pittsburgh, Pennsylvania 15213	1
Stanford University Department of Operations Research Stanford, California 94305 Attn: Prof. G. B. Dantzig	1
Chief of Naval Research (Code 436) Department of the Navy Arlington, Va. 22217	1
Science Librarian Kresge Library Oakland University Rochester, Michigan 48063	1

University of Connecticut	Copies
Department of Statistics Storrs, Connecticut 06268 Attn: Prof. H.O. Posten	1
	1
ARCON Corporation Lakeside Office Park North Avenue at Route 128 Wakefield, Massachusetts 01880 Attn: Dr. A. Albert	1
Stanford University Department of Statistics Stanford, California 94305 Attn: Prof. H. Chernoff	1
Yale University Department of Statistics New Haven, Connecticut 06520	1
Rutgers-The State University Statistics Center New Brunswick, New Jersey 08903 Attn: Prof. H.F. Dodge	1
Yale University Department of Statistics New Haven, Connecticut 06520 Attn: Prof. F. J. Anscombe	1
Purdue University Division of Mathematical Sciences Lafayette, Indiana 47907 Attn: Prof. S.S. Gupta	1
Cornell University Department of Industrial Engineering Ithaca, New York 19850 Attn: Prof. R.E. Bechhofer	1
Mrs. Barbara Eaudi Univ. Program Coordinator, B.E. NASA Johnson Space Center Houston, TX 77058	1

Professor Geoffrey S. Watson, Chairman	Copies
Department of Statistics Princeton University Fine Hall Princeton, N. J. 08540	1
Stanford University Department of Mathematics Stanford, California 94305 Attn: Prof. S. Karlin	1
University of Sheffield Department of Probability and Statistics Sheffield 10, ENGLAND Attn: Prof. J. Gani	1
University of California Operations Research Center Institute of Engineering Research Berkeley, California 94720 Attn: Prof. R.E. Barlow	1
Stanford University Department of Statistics Stanford, California 94305 Attn: Prof. H. Solomon	1
Applied Mathematics Laboratory Naval Ships Research Development Center Washington, D.C. 20007	1
Systems Analysis Division Room BE760, Pentagon Washington, D.C. 20350 Attn: Mr. A.S. Rhodes, Op-964	1
Department of Statistics University of North Carolina Chapel Hill, North Carolina 27515 Attn: Prof. M.R Leadbetter	1
Southern Methodist University Department of Statistics Dallas, Texas 75222 Attn: Prof. D.B. Owen	1
Israel Institute of Technology Technion Haifa, ISRAEL Attn: Prof. P. Naor	

	Copies
Office of Gifts & Exchanges Library Texas A&M University College Station, Texas 77843	2
Archives Texas A&M University College Station, TX 77843	1